UNIVERSITY OF TURKU
Turku School of Economics

Stock Market Prediction with Long Short-Term Memory Networks: A Multi-Market Performance Analysis

Master's thesis
in Finance

Author:
Veeti Härkönen

Supervisor:
Prof. Luis Alvarez Esteban

13.5.2025
Turku

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the Turnitin Originality Check service.

Master's thesis

Subject: Finance
Author(s): Veeti Härkönen
Title: Stock Market Prediction with Long Short-Term Memory Networks: A Multi-Market Performance Analysis
Supervisor(s): Prof. Luis Alvarez Esteban
Number of pages: 65 pages + appendices 3 pages
Date: 13.5.2025

This thesis investigates the use of Long Short-Term Memory (LSTM) models in financial trading, assessing their performance in both individual stock level and portfolio allocation across three markets: the New York Stock Exchange (NYSE), India's National Stock Exchange (NSE), and Finland's OMX Helsinki (OMXH). The study evaluates whether LSTM-driven strategies can outperform passive benchmarks such as buy-and-hold and equally weighted portfolios.

Results show that the LSTM model-based trading algorithm underperformed buy-and-hold strategies in individual stock trading across most markets, mainly due to limited predictive accuracy and difficulty capturing sustained trends. However, when applied to portfolio allocation, specifically through Rank-Based Allocation (RBA) and Exponentially Rank-Based Allocation (ERBA), the models consistently outperformed the equally weighted benchmark. This suggests that while the LSTM model struggled with absolute price prediction, it was more effective at ranking stocks by relative performance. When factoring transaction costs into the active allocation methods, costs must remain below 0.005%-0.01%, depending on the market and method, to outperform the passive benchmark.

The study also examines whether market characteristics like size and economic development influence the model's outcomes. Although predictive accuracy tended to decline in the larger and possibly more efficient NYSE stock market, allocation strategies performed best in it, indicating that prevailing market trends may matter more than structural factors.

Lastly, the research explores the effect of signal sensitivity, finding that increased trading signal frequency generally improved prediction accuracy, particularly in OMXH and NSE, though its impact on returns remained limited.

In conclusion, while LSTM-based models are not consistently effective for individual stock trading, they show promise in portfolio allocation. The study highlights limitations such as a short backtesting period and exclusion of real-world events like slippage, recommending that future work explore enhanced model configurations, alternative allocation methods, and longer testing horizons.

This study utilised AI-based tools: Grammarly was used to check and improve the language of the thesis, while ChatGPT assisted with coding tasks, debugging, and explaining programming concepts related to the trading models.

Key words: Long Short-Term Memory, Neural Network, Trading Algorithm, Rank-Based Allocation

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Background and motivation

Algorithmic trading has been growing in popularity with all financial instruments and accounts for 30 % to 80 % of all transactions in many markets worldwide (Teall, 2018, 3). Algorithmic trading refers to trading where sophisticated algorithms are used to automate all or some part of the trading process (Treleaven et al., 2013, 76). Algorithmic trading can include decision-making, learning, reasoning and dynamic planning (Treleaven et al., 2013, 76). In some cases, it is defined as automated trading where the algorithm collects the data and makes decisions and transactions on its own without any human contribution (Seyfert, 2018, 197–198). This enables systematic trading strategies and reaction speeds that humans can't compete with.

The impact that algorithmic trading has on the markets is vast. Algorithmic trading increases the efficiency and liquidity of the markets, and it can also increase volatility during uncertain periods (Dubey et al., 2022, 1–3). But maybe the most prominent impact, and even partly what causes the mentioned impacts on the markets, is the speed at which algorithmic trading allows investors to make transactions (Dubey et al., 2022, 1–3). Being the first to capitalise on market trends and movements maximises the possible returns, which is why investors compete with the fastest algorithms. This competition is done through optimising algorithms and developing and implementing new technology. The progress in information sciences and automation technology correlates strongly with the trading speeds at which transactions can be made (Foucault & Moinas, 2018, 1–3). With better and faster technology, the trend of transaction speed only goes upwards.

In addition to transaction speeds, algorithmic trading also increases the speed at which markets are impacted by new information. The increased rate of price adjustments in the markets makes it harder to leverage new information to gain excess returns compared to the average market returns, thus making the markets more efficient. (Frino et al., 2020, 749‑760)

Large institutions can use algorithmic trading to disguise their large transactions by dividing the transaction into smaller orders and making it look like transactions of multiple smaller investors, otherwise, their transactions would be noticed by other investors, and a large buy order would increase the price of the stock significantly (Teall, 2018, 14–16). This kind of algorithmic trading is therefore used to minimise execution risk and the price effect their transaction would have.

With these impacts in mind, algorithmic trading demands active monitoring by regulators to mitigate its potential negative effects and to increase market fairness for all market participants (Dubey et al., 2022, 1–5).

Machine learning in trading algorithms is a way to develop data-driven trading strategies. While fundamental and technical analysis are still widely in use, increased complexity in the stock markets has made data-driven trading more popular. Machine learning enables trading algorithms to adjust the strategy based on market conditions, and even this can be fully automated. The trading algorithm can receive instant feedback and adjust itself accordingly. It doesn't rely on assumptions of the market but only on its observed data when making decisions. When machine learning is applied to trading, it can be used in portfolio optimisation and risk management and in an ideal case, it will improve over time as it evolves by observing the market and the data from it. (Abbracciavento et al., 2022, 1032–1041)

Due to the increase in data-driven strategies and machine learning, the role of data science in finance has become crucial. Financial markets generate vast amounts of diverse and complex data, which financial institutions and investors are trying to utilise to maximise returns. This data offers insights into market trends and transactions, and in order to make use of it, it has to be analysed quickly and thoroughly. The quick analysis enables one to profit from rapid market movements, which makes data science invaluable. The amount of data and the need for quickness demand huge amounts of computing power. Investors often utilise cloud computing in order to handle huge data sets and scale their computational resources. In addition to algorithmic trading, data science is important for financial institutions for their fraud detection, risk management and automation of other processes. (Huttunen et al., 2019, 23–25)

Analysis of the data enables pattern recognition from historical price data, which is traditionally used for technical analysis. Nowadays, investors are utilising machine learning to recognise patterns and make investment decisions based on those patterns. The speed and consistency of the algorithm are crucial for the success of the trading strategy built around pattern recognition (Tsinaslanidis & Guijarro, 2021, 2). This makes the backtesting a crucial part of the strategy, as no one can cancel faulty transactions of the model when they happen almost instantly. Backtesting is also a way to find out which patterns could be used to predict market trends and price movements, and how consistent the results of the algorithm are.

What makes neural networks really powerful tools in finance is their ability to capture the non-linearity of financial data compared to other popular forecast tools, such as linear regression, which

would benefit more from clean datasets with linear relationships (Fadlalla & Lin, 2001, 120). Furthermore, what sets neural networks apart from many other models is their flexibility to work with complex data, which makes them more than ideal to predict the stock market. Novel neural networks with memory capabilities, such as recurrent neural networks, can detect long-term dependencies between variables in time series (Fabbri & Moro, 2018, 143). These properties have made recurrent neural networks popular in finance and interesting to study more deeply.

Although recurrent neural networks themselves have features that make them great at forecasting time series, Long Short-Term Memory (LSTM) networks are more of the focus point for this study. LSTMs are subsets of recurrent neural networks, but they improve upon some problems with vanilla recurrent neural networks we will discuss later in the theory part of the thesis, which is why we will be using them in our forecasting. The key advantage of LSTMs compared to recurrent neural networks is that they can remember previous data when recurrent neural networks would tend to forget it too quickly (Wang et al., 2024, 1–3). This advantage should make it even more accurate with sequential stock series forecasting.

Even though algorithmic trading is well-researched, the research itself faces some challenges. Many of the studies provide evidence of promising results in unrealistic settings, which makes the application to real-world trading uncertain. Even when the evidence points to improvements over the baseline strategies, the difference is not significant for the majority of the research. It can also be assumed that if the new algorithm does perform well in the real world, it would not be published to give away its edge. (Pricope, 2021, 1–17)

Another variable in the performance of trading algorithms can be assumed to be the level of implementation of trading algorithms in each specific market. Information about the influence of competitive trading algorithms in the markets on the generated returns would be especially valuable for large institutions, as they have the resources and the knowledge to develop and implement complex trading algorithms in any markets they wish. Performance of trading algorithms, LSTM-powered models in this study, across different international markets, would therefore be valuable to study, as then investors wishing to use trading algorithms could allocate their resources to the markets with the highest potential for success.

1.2 Purpose of this study

This study aims to evaluate the performance of neural networks in trading algorithms. Evaluation is done by backtesting the developed Long Short-Term Memory (LSTM) neural network model, and

the trading algorithms based on it, with historical price data of multiple individual stocks from Finnish, Indian and US stock markets. The objective for this study is to answer the following research questions:

1. *Are the trading algorithms, based on the LSTM model, able to outperform the benchmark methods?*

The hypothesis is that the LSTM model used in this study could outperform the benchmark strategy of buy-and-hold in individual stock level trading and equal-weighted portfolio on the portfolio level trading. This information would be beneficial for investors as they could use this or other LSTM models to increase the returns of their investments. The buy-and-hold method and the equally weighted portfolio serve as benchmarks as they are passive and widely used strategies in portfolio management, and if an active strategy with the same financial instruments can outperform the passive strategy, in theory, these passive strategies could be shorted to finance the active trading strategy to increase the profits.

2. *Will the size of the market or the developmental stage of the country influence the performance of the LSTM-based trading algorithms?*

In this study, the LSTM-based trading algorithms are tested across three stock markets with different characteristics, and the results are compared to each other in order to find out if these characteristics influence the profitability of the trading algorithms. The hypothesis is that larger and more developed markets would be more efficient, and generating excess returns and accurate predictions would be more difficult than in smaller or less economically developed markets. The three stock markets considered in the study are from the United States, Finland and India.

3. *Is the sensitivity of the signal generation impactful for the results of the LSTM-based trading algorithm?*

The last objective is to explore how the sensitivity of the signals influences the performance of the trading algorithm, used in the individual stock level, across different markets. This information is critical as it defines how the algorithm reacts to price movements in the markets and how frequently the signals are generated. The hypothesis is that the frequency of the signal generation would be a significant variable for the performance of the trading algorithm and the LSTM model, and more active signal generation would be the most optimal for the algorithm.

1.3 Limitations

This study faces some limitations and constraints that might affect the results and conclusions, and which might be good topics for future studies. The first set of limitations comes with the dataset and its availability and coverage. This study uses daily data from *Yahoo finance,* which limits the amount of data available in the API with higher frequencies. The trading algorithm in this study could be considered low low-frequency trading algorithm based on the kind of data it tries to trade with. Future studies could experiment with hourly or even shorter frequency data, and if it influences the results. This would eventually require way more computational resources if the dataset covered the same time frame as in this study. The length of the time frame can also affect the results, as it might not cover all of the relevant market conditions that could appear in the future. This could affect the generalizability of the study.

This study also excludes some real-world features in investing, such as taxes, slippage and execution delays. The results of the model and the trading algorithm might therefore be more positive than they would be when implementing the strategy in the markets. The decision to leave these features out has been made to achieve more generalised results. Taxes are excluded from this study as some countries don't have capital gains taxes, and some do, which would make the results with taxes apply only to investors investing in countries with that specific tax rate. Slippage and execution delays are excluded as this study focuses on the theoretical performance of the trading algorithm, and due to the long-term time period in this study, the effect of those variables would most likely be minimal.

1.4 Structure of the thesis

This thesis is divided into 5 sections. The first one introduces the topic of the study, trading algorithms and machine learning. In the introduction, the background and motivation for this study are discussed, and the purpose is also defined. The study also comes with some limitations that are introduced and justified. The introduction serves as a mean to scratch the surface of the topic and to build the narrative the study follows throughout the thesis.

The second section of the study discusses the theoretical framework relevant to this study. It includes theory about market efficiency, requirements for markets to enable the success of trading algorithms working with historical data, theory about machine learning and neural networks and also previous studies related to this field of study. The theoretical framework should help with understanding the methodology and mechanisms used in this study.

The third section introduces the data used in this study, how it is collected and why it is selected and also the methodology used to gather the results to answer all of the research questions. The introduction of methodology seeks to enable future studies to duplicate the results by copying the methods used in this study. This would make it possible to build upon this study for future research.

The fourth section deals with the results of the study. The results are showcased, discussed and evaluated. This section answers the research questions and attempts to explain why the results are what they are.

The last section discusses the findings of this study and summarises the key findings. It concludes the most important information this study was able to generate and what could be some of the possible topics for future studies that could be built upon the findings of this study.

2 Theoretical framework

2.1 Predictability of stock markets

2.1.1 Random Walk Theory

Although the random walk theory (RWT) was first proposed by Louis Bachelier in 1900, it was made famous by Burton Malkiel in 1973. Malkiel argues for the unpredictability of stock markets. RWT implies that price movements in stock markets are unpredictable, and the use of historical information will not help with investment decisions (Malkiel, 1973, 17–18). This means that investment decisions made randomly would generate, on average, the same returns as decisions made based on various kinds of financial analysis. This is because, according to RWT, historical price movements or trends do not influence the direction or size of future price movements (Malkiel, 1973, 17–18). This unpredictability indicates that yesterday's price is not connected to today's price. Andrew W. Lo and A. Craig MacKinlay wrote a book in 1999 as a counter to RWT. It's a collection of theories and essays on predictability and patterns of stock markets. These essays attempt to disprove the RWT through statistical and visual proofs as well as theories around outperforming the market.

In 1965, Eugene Fama investigated the statistical properties of stock markets and argued for RWT. RWT in stock prices includes two main hypotheses, which are that price movements are independent of one another and that price movements create some probability distribution (Fama, 1965, 35). Independence means that the probability distribution of upcoming price movements is independent of previous price movements (Fama, 1965, 35–36). Based on empirical analysis, price movements of stocks did not follow any significant patterns that could be used in predicting future price movements (Fama, 1965, 37). Fama also emphasised the effect of new information on stock prices, laying the groundwork for his future studies of the efficient market hypothesis (Fama, 1965, 38–39). The article implies that investment strategies based on technical analysis are unable to generate excessive returns consistently, and even fundamental analysis is unlikely to be effective as new information has been priced into the stock value. These results would highly support passive investment strategies, such as buy-and-hold strategies, compared to active trading strategies, such as the trading strategy tested in this study.

Even though markets may behave unpredictably, some factors increase the predictability of markets. Lo and MacKinlay (1988) found short-term positive serial correlation of returns. This information implies that well-performing indices will continue to perform well in the short term. In addition, Fama and French (1988) and Poterba and Summers (1988) identified long-term mean reversion. This means

that in the long term, stocks tend to revert to their historical average, implying that well-performing stocks will likely perform poorly in the long term. These findings challenge the thought that stock returns would be independently distributed as Fama assumed in his article. In addition to these findings, the authors propose that while linear relationships might be hard to find between stocks, they often show non-linear dependencies, which is inconsistent with RWT.

These inconsistencies with RWT might enable the use of trading algorithms based on analysing historical data of stock markets. Especially trading algorithms based on neural networks, which are particularly good at identifying nonlinear relationships in data.

2.1.2 Efficient Market Hypothesis

The Efficient Market Hypothesis (EMH) continues the random walk theory (RWT) where information is not seen as an advantage for investing, as all information about stocks is already priced into the stock prices. From this starting point, Eugene Fama refined the concept of RWT into EMH (1970). Fama defined efficient markets and categorised them into three different efficiency levels based on the type of information priced into the stocks (Fama, 1970, 383). Fama also defined the expected price of stock given the information in the markets with the following formula:

$$E(p_{j,t+1} \,|\phi_t) = [1 + E(r_{j,t+1}|\phi_t)]p_{j,t}$$

In the formula above the E denotes the expected value operator and p is the price of the stock j at the time t. $p_{j,t+1}$ is the value of the stock in the next t and $r_{j,t+1}$ is the return of the stock in percentage between periods t and t+1 and lastly the ϕ_t denotes the information about the stock at the time t, which is assumed to be priced into the stock. (Fama, 1970, 384)

Based on this formula, the accuracy of the prediction of the price of the stock in the next period increases as the information about the stock increases and all the information available is included in the expected return and expected price of the stock. The formula also defines what efficient markets mean; the new information in the markets is priced into the stock instantaneously in the next period. Below is the visual representation of the difference between efficient and inefficient markets.

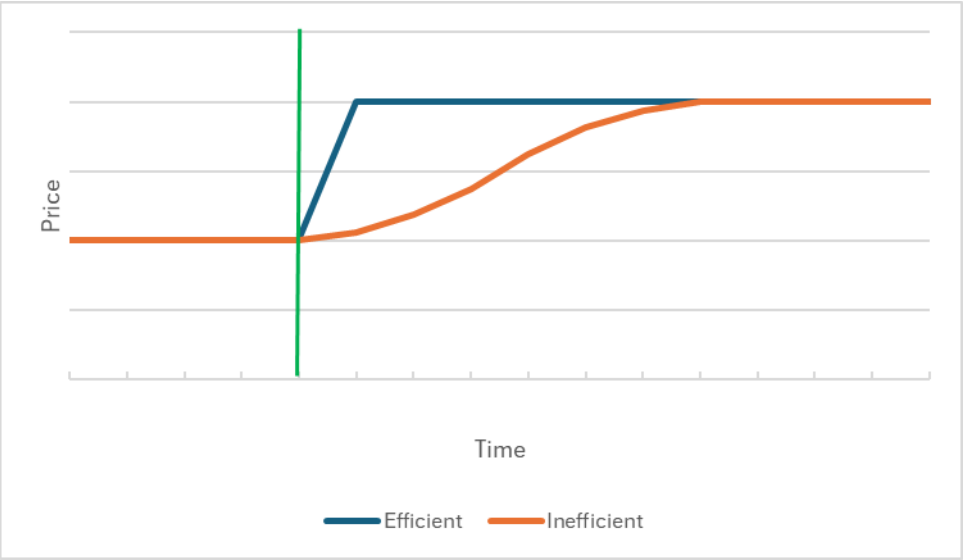

Figure 1 Impact of information in efficient and inefficient markets

In the chart, the green line marks the time that new information is released in the markets and in the efficient markets shown with the blue line, the price fixes itself to the new level in the next period while in the inefficient markets shown with the orange line, the impact of information is slower, indicating that the information takes time to be priced into the stock.

Fama divides efficient markets into three categories based on the type of information. The three categories are weak, semi-strong, and strong form efficient markets.

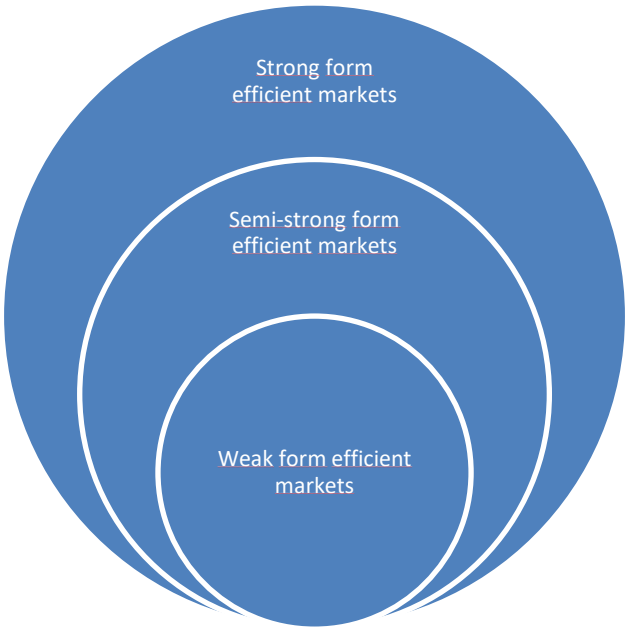

Figure 2 Efficient market hypothesis Venn diagram

The Venn diagram above illustrates the relationship between these three efficient markets. Weak form efficient markets are included in semi-strong efficient markets, and semi-strong efficient markets are

included in strong-form efficient markets. The weak form efficient markets are defined as markets where historical prices and returns are reflected instantly in the stock prices (Fama, 1970, 383). For investors, this implies that in order to generate returns higher than the expected market returns, they have to have information about the stocks beyond the historical prices. This information can include financial reports and news about companies that are publicly available to all investors. In the semi-strong form of efficient markets, even if information is included in the price of the stock (Fama, 1970, 383). Therefore, investors have to obtain insider information that is not available to other investors to obtain higher-than-expected market returns. In strong-form efficient markets, even insider information is priced into the stock, which means that there is no way to gain an edge using any kind of information in the markets (Fama, 1970, 383).

For markets to be truly efficient, Fama lists three conditions enabling market efficiency: no transaction costs, all information is available for all investors without costs, and all investors agree on the implications of the information for the stocks (Fama, 1970, 387–388). Markets where all these conditions are met are fully efficient. Unfortunately, in the real world, these conditions are not realistic. Despite this, markets can be efficient enough to be considered efficient.

EMH has been tested in multiple studies with various results. Fama introduces empirical work relating to EMH in his original article introducing EMH, and the results generally support weak and semi-strong form efficiency in the market. This indicates that stock prices reflect the historical price data and publicly available information in the markets. In 1991, Fama continued the work related to EMH by reviewing more empirical work made to test market efficiency. The three main topics are event studies, private information and return predictability. Fama presents event studies as the cleanest method to provide evidence on market efficiency. Event studies can study the impact of new information on the markets. The provided evidence suggests that generally stock prices adjust quickly to new publicly available information, supporting weak and semi-strong efficiency in the markets (Fama, 1991, 1607). Research on private information is less common, but the evidence from studies suggests that investors in companies or pension and mutual fund managers are able to generate abnormal returns, suggesting the use of private information, and it is beneficial in investing (Fama, 1991, 1607–1608). This implies that markets are not strong-form efficient. The studies about return predictability, on the other hand, indicate positive autocorrelation in the short term and negative autocorrelation in the long term. Intuitively, this would be evidence against EMH and weak form efficiency, but Fama suggests that this is mainly resulting from the selection of data, and the results are not statistically as significant when some of the market anomalies are excluded.

Despite this being Fama's stance on the matter, a lot of research suggests otherwise. The previously mentioned John Y. Campbell, Andrew W. Lo and A. Craig MacKinlay discuss the return patterns in the market, providing evidence for market predictability and against EMH and weak form efficiency. Another study from Andrew W. Lo and A. Craig MacKinlay published in 1988 provides evidence against RWT based on the results from a simple specification test. The results are not used to refute EMH, but they do impose restrictions that the price formation paradigms have to explain for markets to be efficient (Lo & MacKinlay, 1988, 27–28).

Burton G. Malkiel (2003) discusses the various methods of observing patterns in the markets, providing evidence for market inefficiencies. Some of these inefficiencies are market momentum, long-run return reversals, seasonal and day-of-the-week patterns, and patterns in valuation parameters (Malkiel, 2003, 61–64). Momentum refers to a behavioural finance concept of bandwagon effect, where investors are drawn to the market where returns are high (Malkiel, 2003, 61–62). Another, more rational explanation for the effect relates to the efficiency of the understanding of new information in the markets (Malkiel, 2003, 61–62). If the consequence of the information is not instantly understood by market participants, the effect will be gradually priced into the stocks, which would be an example of inefficiency in the markets. Long-run return reversals, on the other hand, refer to negative autocorrelation of the stock price in the long run (Malkiel, 2003, 63–64). One explanation for this is another inefficiency of understanding the market information, more precisely, overreaction in the prices due to new information being published (Malkiel, 2003, 63–64). This can be caused by investors' optimism or pessimism about future price developments, and it leads to reversals when the real outcome of the information is observed (Malkiel, 2003, 63). This inefficiency in the markets is used by contrarians, who are investors who bet against ongoing market trends (Malkiel, 2003, 63). Seasonal and day-of-the-week patterns are patterns that can be observed during certain periods (Malkiel, 2003, 64). For example, market indices have been documented to be unusually high during the first two weeks of January and out of all weekdays, Mondays seem to generate the highest returns on average (Malkiel, 2003, 64). And lastly, inefficiencies can be found in valuation parameters such as Price-Earnings and Dividend-Price ratios (Malkiel, 2003, 64–66). The Price-Earnings ratio has a clear negative correlation with future returns, and the Dividend-Price ratio has a positive correlation with future returns (Malkiel, 2003, 66). Having said that, despite this relationship between the ratios and stock prices, it is suggested that investors should be cautious when basing decisions on these ratios, for example, the S&P 500 had high Price-Earnings and low Dividend-Price ratios from 1987 to 2002, but despite this, the index performed well during the period

(Malkiel, 2003, 67). These four mentioned inefficiencies are against the EMH and can, in some cases, be used in investment decisions.

EMH creates certain challenges for active traders in the markets. EMH posits that all available information is priced into the stock prices, making it difficult to outperform the market through active management of assets. This also implies that the use of algorithmic trading, which utilises technical information of the markets, such as within this study, would not be able to generate excess returns. Therefore, if the trading algorithm in this study were to outperform the markets, it could be used as an argument against EMH.

2.2 Algorithmic trading

A trading algorithm is an automated system for executing transactions of financial instruments (Kissell, 2013, 1–2). Trading algorithms can trade all kinds of financial instruments such as stocks, bonds and currencies. Algorithmic trading (AT) is done by specifying mathematical instructions for computers to execute transactions according to these instructions, and it is used widely by mutual, index, pension, quantitative, and hedge funds (Kissell, 2013, 1–2). The wide use of AT has many significant effects on financial markets. The increased amount of transactions increases the liquidity in the markets as they are working almost as market makers, buying and selling financial instruments constantly, enabling quick trades for manual investors (Gomber et al., 2011, 2). AT also contributes to price discovery and market efficiency by increasing the amount of buy and sell orders, bringing the difference between them, the spread, closer (Gomber et al., 2011, 2). Even though AT has positive impacts on market quality, it is not its sole purpose in the markets. AT is mainly used for maximising returns or controlling execution costs and market risk (Gomber et al., 2011, 52, 60). The object of AT depends on the investor, large institutional investors may use AT as a method for splitting up their large buy orders into smaller ones so the effect of the order on the price would be minimised. Speculative traders, on the other hand, utilise AT in maximising returns (Chakravarty & Sarkar, 2020, 76). In this study, we focus on the speculative trading side of AT. It is made by exploiting imbalances and imperfections in the markets (Chakravarty & Sarkar, 2020, 76). These imbalances and imperfections in the markets usually last for a very short period, which is when the benefits of AT come into play. AT can enable investors to act quickly to catch opportunities to generate returns (Chakravarty & Sarkar, 2020, 76). High-speed transactions in the market require speed from information collection and hardware. This means that investors utilising AT for speculative trading have to minimise latency in their data collection and increase their computing power to enable fast decision-making from their computers (Chakravarty & Sarkar, 2020, 76).

Because AT relies on imperfections in the markets, it is assumed that for AT to generate excess returns consistently, markets should be inefficient, which is against the efficient market hypothesis (EMH).

Despite the popular belief in EMH, there are a large number of investors utilising a wide range of different algorithm-based investment strategies. One of them relies upon statistical arbitrages through pairs trading. This includes finding two securities that have historically similar price movements (Krauss, 2017, 2). The spread between the prices of these two securities is then monitored, and when the spread increases, the strategy is to buy the loser and short the winner in order to profit when the spread reverts to its historical mean (Krauss, 2017, 2). This same concept can also be applied to portfolios with multiple securities. Pairs trading is a widely studied strategy with multiple different approaches, such as distance, cointegration and time series approach (Krauss, 2017, 3). The distance approach is the most common approach to pairs trading, and it involves using mathematical distance metrics to find securities that move similarly to each other, and when prices diverge above a certain threshold, the approach suggests trading signals (Krauss, 2017, 3). This approach has had positive results, for example, with a study conducted by Gatev et al. (2006). The cointegration approach, which is more statistically robust, involves looking into the long-term relationship known as cointegration between two securities, and it assumes that if they diverge, they will revert back to their normal relationship (Krauss, 2017, 26). This approach has had positive results by Caldeira and Moura (2013). Lastly, the time series approach focuses on trading, instead of finding the pairs and assumes that the pairs are already chosen and generates optimised signals based on time series analysis (Krauss, 2017, 3). Also, this approach has had positive results in trading energy futures in the study by Cummins and Bucca (2011). Another popular use case for algorithmic trading is called momentum trading. Simply put, it assumes that securities that generate returns keep generating returns in the future (Martin, 2023, 1–2). It therefore involves following trends in the markets, buying stocks that are going up and selling stocks that are going down. A lot of research has been done on this method of trading, with positive and negative results. An example of a study with positive results is a study by Foltice and Langer (2015).

On top of the strategies mentioned above, algorithmic trading can also utilise machine learning and artificial intelligence approaches. The importance of machine learning in finance has been increasing, especially in the last few years (López De Prado, 2020, 1). This is due to the complexity of the economic environment, which classical statistical tools are not able to interpret fully (López De Prado, 2020, 1–2). The increased importance of machine learning for finance can be observed in the widened range of use cases for machine learning used in asset management. This trend is predicted to continue as computer processing power increases and algorithms become more complex and

efficient (López De Prado, 2020, 1). Research on machine learning and artificial intelligence approaches in finance has been on the rise, and one interesting take on it has been the comparison of these approaches to classical statistical methods. One example of this is a study conducted by Venkatarathnam et al (2024), which analysed data from Indian stock markets using a classical statistical method, ARIMA and compared its results to results from neural network-based models. Even though ARIMA can provide useful insights into the market data, especially in the short term, neural network models such as Recurrent Neural Network and Long-Short Term Memory models were more accurate in certain conditions, such as longer-term prediction and when capturing more complex patterns on the markets (Venkatarathnam et al., 2024, 173).

2.3 Neural networks in finance

Artificial neural networks (ANN) are often compared to natural neurons in the brain as they both consist of neurons connected to each other and communicate via sending signals between them. Despite this similarity, they are quite different. While natural neurons communicate with binary spikes at different frequencies, ANNs can communicate with actual numbers. ANNs are also structured in layers, while natural neurons are structured in more complex ways. The training is distinctly different with ANNs and natural neurons, as ANNs are usually trained with supervised training, where these neurons learn their weights, while natural neurons can evolve without getting direct correct answers. Because of these differences, some researchers prefer the term deep learning, but it is essentially the same as neural network. (Koehn, 2020, 31)

The beginning of ANNs was in 1943 when Warren McCulloch and Walter Pitts wrote a research paper about how neurons could function. In the paper, neurons were binary devices that could only signal through all-or-nothing functions. Frank Rosenblatt conducted a more promising ANN study in 1958. The study introduced perceptrons, which were structured in one layer and could learn basic mathematical functions such as Boolean, AND, and OR.

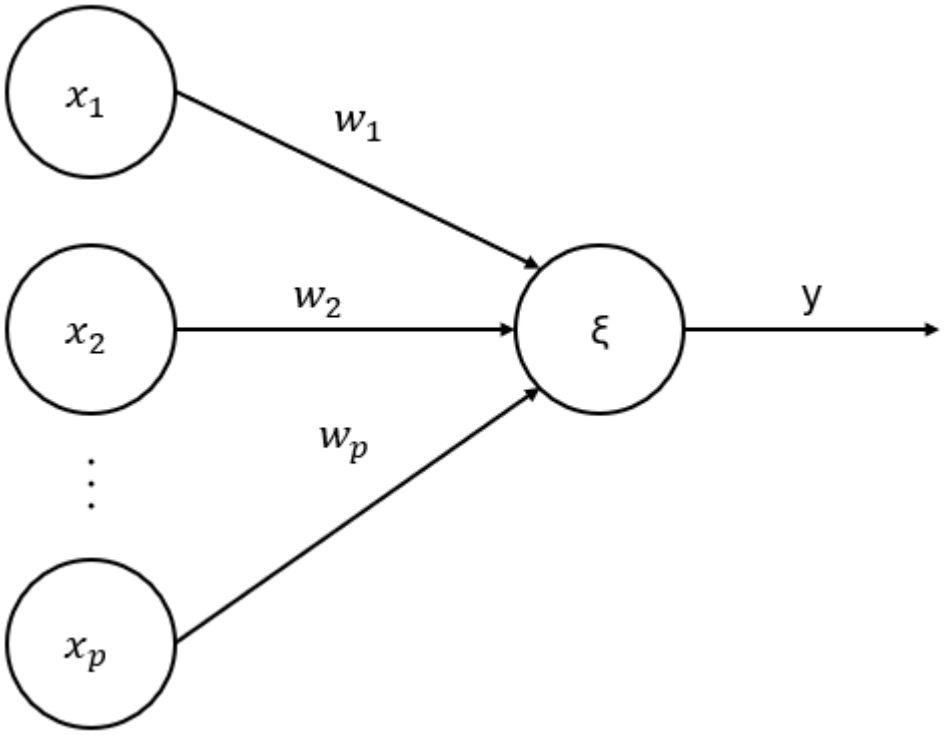

Figure 3 Perceptron model

The perceptron model above consists of input neurons x_1, x_2,…, x_p which represent features of the data inputted into the model, and each of these input neurons has its weight *w* that determines the importance of each input. All weighted inputs are summed up in the neuron ξ, which follows the equation below.

$$\xi = \sum_{i=1}^{p} x_i w_i + \theta$$

The weighted inputs are summed up with the bias θ using the formula above and the result is passed through the activation function below.

$$y = s(\xi) = \begin{cases} 1, & \xi \geq 0 \\ 0, & \xi < 0 \end{cases}$$

The activation function receives the sum ξ and passes through value 1 if ξ is equal to or larger than 0, and 0 if ξ is less than 0. The model, therefore, only returns binary values. Weights w and bias θ are determined in the model through training. This simple model is the basis for the more complex neural network models developed later on. (Bishop & Nasrabadi, 2006, 192–193)

With more advanced neural network models and increased computational power, came more diverse applications of neural networks such as neural network applications in finance (Aldridge & Avellaneda, 2021, 15–16). Previously, neural networks were a novel tool in finance since before, the cost of developing neural networks was higher than its benefits (Aldridge & Avellaneda, 2021, 15–16). Nowadays, there are multiple use cases for neural networks in the financial field (Aldridge & Avellaneda, 2021, 17). One of the most popular use cases and most relevant for this thesis is time series prediction. Time series prediction in the stock markets is considered to be one of the most difficult challenges in time series prediction (Bao et al., 2017, 1). One type of neural network that is commonly used in time series prediction is recurrent neural networks and their variants, like the long short-term memory model (Bao et al., 2017, 2). This model has been chosen to be studied in this thesis based on its popularity and supposed effectiveness.

2.3.1 Recurrent neural networks

Recurrent neural networks (RNN) are neural networks that can be used to process sequential data (Goodfellow et al., 2016, 367–368). RNNs use connections to feed neurons' outputs into themselves, enabling the processing and handling of sequential data and its temporary dependencies (Goodfellow et al., 2016, 367–368). This loop that provides feedback internally makes RNNs especially good at time series prediction tasks. This internal loop can be visualised with the graph below without output nodes.

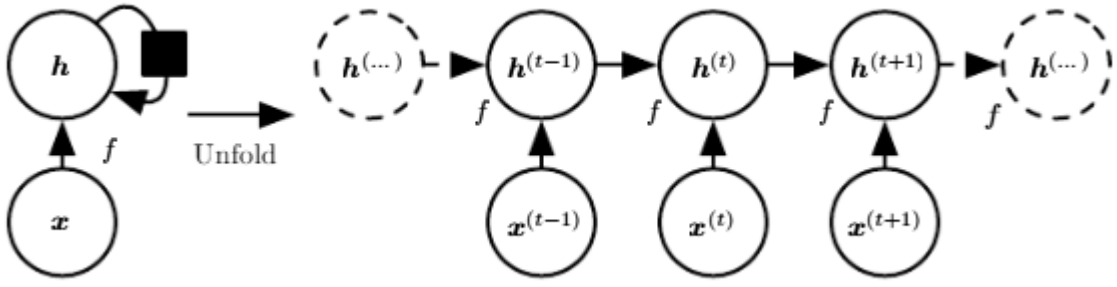

Figure 4 Recurrent Neural Network internal loop (Goodfellow et al., 2016, 370)

The left side of the figure above illustrates the internal loop or RNNs without the output. The x is the input, and h is the state of the RNN which is passed to the next node. The black box conveys the lag

of one, showcasing that the node receives two inputs, the input *x* and the state *h* with a lag of one. The same figure is showcased in an unfolded manner on the right, emphasising the model's rolling fashion where the *h* is passed from one node to another while receiving an input *x* in all time steps. In addition, an output would usually be produced from each time step. The main difference between RNNs and feedforward neural networks (FNN) is that FNNs do not have the state *h* so the information flows directly from input to output through the network. FNNs are mostly used in static tasks such as image recognition and RNNs are better suited for sequential tasks such as time series forecasting (Goodfellow et al., 2016, 164). The state *h* can also be referred to as a hidden state and it can be described with the following formula:

$$h_t = f(h_{t-1}, x_t;\ W)$$

In the formula the h_t conveys the hidden state at the time of *t* and it can be expressed as a result of a function *f* that utilises the value of the input x_t and the hidden state h_{t-1} at the time of *t* – 1. The *W* is the weight matrix defining the weights of the function that can also be expressed the following manner:

$$h_t = f(W_h h_{t-1} + W_x x_t + b_h)$$

The formula above showcases the structure of the function *f* and also includes the bias term *b* that is also used in the formula defining the output *o*:

$$o_t = g(W_o h_t + b_h)$$

Therefore, the output o is defined using the function g, which includes the weight matrix W and the bias term b.

The architecture of RNNs enables them to remember previous information about the data sequence, and the information is stored in a unit called a hidden state (Hindarto, 2023, 2539). Due to this, RNNs are able to detect temporal dynamics and patterns in data, which is especially beneficial in time series analysis (Lipton, 2015, 2). Despite being efficient in time series analysis, RNNs possess a quality that makes them weak in long-term predictions. While the hidden state enables RNNs to remember past information and be efficient in time series analysis, it also causes events called exploding and vanishing gradients (Chung et al., 2014, 2–5). This is caused by the property of RNNs where they pass information from one step to another through hidden states, which are updated using the weights *W* from the formulas above. When RNNs update their hidden states or memory, they use weights as multipliers. These weights are able to cause either exploding or vanishing gradients based on their

values. If for example, these weights are too small, like 0.5, repeated multiplication using these weights causes the so-called gradient to fade towards zero or vanish and on the contrary, if the value is too large, like 2, repeated multiplication causes the gradient to increase uncontrollably or to explode. This property of RNNs makes them inefficient with long-term predictions, as RNNs have more steps where the weights are used as multipliers, increasing the impact of vanishing or exploding gradients. This problem with RNNs has a couple of solutions, such as limitations to the values or more balanced weights, but to fix these concerns, there are more developed versions of RNNs, such as the Long Short-Term Memory model, which is also going to be used in this thesis.

2.3.2 Long short-term memory

Long Short-Term Memory (LSTM) is a type of RNN designed to address traditional RNNs problems by introducing a more complex structure that mitigates the impact of vanishing and exploding gradients. The hidden state that retains the memory of past information and causes the gradient problem with RNNs is replaced by four components: cell state, forget gate, input gate and output gate. These components control the flow of information, what information is added, what is retained and what is passed on to the next node of the network. This structure specifically tackles the deficiencies of RNNs' hidden states and improves the performance of long-term time-series predictions by capturing long-term dependencies more accurately. (Hochreiter & Schmidhuber, 1997, 1–2)

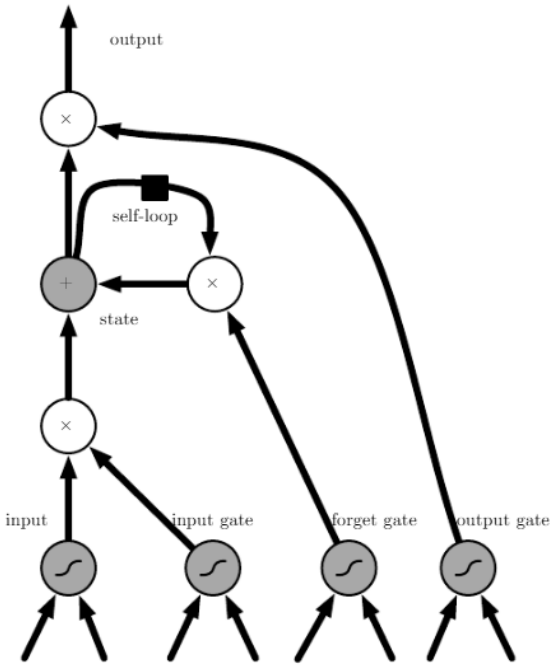

Figure 5 LSTM cell (Goodfellow et al., 2016, 405)

The structure of the LSTM network is visualised in the figure above. Like in RNNs, the structure contains input and output nodes and the hidden state node, which contains the past information. In addition, LSTM networks have three gates that control the information flow of the network. These include input, forget and output gates. Their names refer to which information they control: the input gate controls how much of the information is taken into the network as an input, the forget gate controls which information is retained for the next iteration and the output gate controls which information is passed through to the next LSTM cell. (Goodfellow et al., 2016, 405–406)

The most impactful part of LSTMs is the state unit, which stores the memory in the cell. The state unit is controlled by the forget gate. Forget gate determines the weight of how much of the memory is retained and forgotten in the state node. The weight is a value from 0 to 1 and utilises a sigmoid unit in its calculation.

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right), \qquad \sigma(z) = \frac{1}{1+e^{-z}}$$

The equation above is used to calculate the value of the forget gate unit $f_i^{(t)}$ in the time t and cell i. σ is the sigmoid unit which is used as an activation function to ensure that the output values are

ranging between 0 and 1. b_i^f is the bias term which can, for example, be set to 1 at the start of the training for the model to remember more information. $U_{i,j}^f$ is the weight matrix used in combination with input vector $x_j^{(t)}$ to allow forget gate to consider new data in its decision of what to forget. $W_{i,j}^f$ is a weight matrix which is used with the vector of past outputs $h_j^{(t-1)}$ to decide which past information is still relevant. The forget gate is used in the calculation of the LSTM cell's hidden state $s_i^{(t)}$:

$$s_i^{(t)} = f_i^{(t)} s_i^{(t-1)} + g_i^{(t)} \sigma \left(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} h_j^{(t-1)} \right)$$

This formula is used to store long-term information across time steps. It combines the new information with the previous cell state. The previous cell state is denoted as $s_i^{(t-1)}$ and the forget gate $f_i^{(t)}$ determines how much of its information is retained in the cell's memory. The input gate $g_i^{(t)}$ determines how much of the new information is added to the cell's memory, and its calculation is similar to the calculation of the forget gate:

$$g_i^{(t)} = \sigma \left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)} \right)$$

The components of the formula are the same as before, but with the input gates' own parameters. The current output $h_j^{(t)}$ is calculated using the formula:

$$h_j^{(t-1)} = \tanh\left(s_i^{(t)}\right) q_i^{(t)}$$

The formula includes a hyperbolic tangent function, which scales the cell state $s_i^{(t)}$ values to be between -1 and 1. The output is controlled by the output gate $q_i^{(t)}$.

$$q_i^{(t)} = \sigma\left(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + \sum_j W_{i,j}^o h_j^{(t-1)} \right)$$

Output gate $q_i^{(t)}$ decides how much of the cell state is used as an output. It has a formula similar to that of other gates, with its parameters.

The unique architecture of LSTMs allows them to overcome the deficiencies of RNNs, such as vanishing gradient and exploding gradient, thus making LSTMs more accurate in capturing long-term dependencies within data sequences (Hochreiter & Schmidhuber, 1997, 23). LSTMs have demonstrated their ability in various data sequence tasks such as speech recognition and machine translation (Greff et al., 2017, 1). This sequence processing ability allows LSTMs to be used flexibly in various applications, including time series forecasting, which is the most relevant ability of LSTMs for this study.

2.4 Previous studies

As within this study, LSTM is generally a widely recognised tool for time series forecasting in finance. Its performance has been under multiple researchers with positive results, supporting its use in financial applications. One example of this research is an academic article by Fischer and Krauss (2018). Fisher and Krauss attempt to forecast the S&P 500 index using the LSTM network while comparing the results to other models such as random forests, logistic regression and other deep neural networks. The LSTM model achieved an average daily return of 0.46% and a Sharpe ratio of 5.8, outperforming other comparable models significantly (Fischer & Krauss, 2018, 17, 28). The effectiveness was highest from 1992 to 2009, and from 2010, the excess return generated by the LSTM model diminished (Fischer & Krauss, 2018, 19). It is suggested that this is due to changes in market dynamics or increased competition in the markets, some possibly using similar methodologies of trading (Fischer & Krauss, 2018, 19–20). The trading algorithm is based on LSTM, choosing stocks in which to invest. The trading is thus based on LSTM, creating an optimised allocation instead of comparing returns on trading with individual stocks. Another relevant example of research on LSTM networks in finance is an article written by Yadav et al. (2020). The study discusses the optimisation of LSTM networks while using them to forecast data in Indian stock markets. Some variables in the study were the state or memory of the model and the number of hidden layers it had. Valuable insights that were found were that the model performed better with more data included in its state or memory compared to a model that reset its state between batches of data points (Yadav et al., 2020, 2099).

Another insight was in regards of the hidden layers of the LSTM model, an optimized number of hidden layers was found to be two, indicating that a more complex model was able to detect dependencies in the data more efficiently than simplistic models and increased amount of layers did not further significantly increase the accuracy of the model (Yadav et al., 2020, 2099). In general, with these insights, the study suggests that LSTM networks are able to be used to forecast stock data in the Indian markets.

Several other studies have explored both LSTM applications in stock markets and the use of rank-based allocation methods. Nelson et al. (2017) predicted price movements of stocks from the Brazilian stock index IBovespa using an LSTM-based prediction model. The results were positive, and the model mostly outperformed the baselines, which were other popular machine learning models. This supports the performance and superiority of the LSTM models compared to many other, more traditional prediction methods. A more general review of the use of LSTM models was conducted by Sezer et al. (2020). They observed that LSTM models were the most popular among all the deep learning models (Sezer et al., 2020, 49). In addition, they concluded that in most of the studies which compare deep learning models, such as LSTM, to machine learning models, deep learning models were superior (Sezer et al., 2020). LSTM models have also been applied to predict stock rankings by Zhang & Tan (2018) where the model uses raw stock data from Chinese stock markets, including open, close, high, low and volume, as input and outputs the ranking of the stocks based on their predicted performance in the future. Stock trading done based on this ranking method outperformed the statistical and machine learning based methods. Another study on LSTM-based ranking methods has been done by Sebastian & Tantia (2024). The study included the prediction of stock prices of stocks from Indian stock markets and ranking stocks based on the calculated predicted returns (Sebastian & Tantia, 2024, 927). The predicted top performers are then used to create a portfolio, and the returns are compared to the NIFTY50 stock index (Sebastian & Tantia, 2024). The results indicated that the portfolios formed based on the method were able to outperform the benchmark index, however, the study did not include transaction costs, thus not providing reliable results for the real-world use of the method (Sebastian & Tantia, 2024, 935-939). This lastly mentioned study matches up the most with the methodology used in our study in regard to the prediction of stock prices and the use of allocation methods introduced in the following chapter but in contrast to the methodology used by Sebastian and Tantia, we introduce transaction costs for the allocation methods to determine their effectiveness in a more realistic setting.

3 Data and methodology

3.1 Dataset

This study aims to observe the effectiveness of neural network-based trading algorithms across three different stock markets. These three stock markets have been chosen based on the size and developmental stage of the countries they are in. The three chosen markets are the US, Finnish and Indian stock markets. The US stock market is the largest in the world and also considered to be the most developed, which is why it is used as a benchmark for other stock markets in this study. Finland, on the other hand, is a comparatively small country with a smaller stock market, while Finland's economy is still considered developed, which is why its stock markets are used as a proxy for observing the effects of the size of the markets for the effectiveness of the trading algorithms. Lastly, India is the largest country in the world based on population, but its economy is considered to be less developed than the economies of the US and Finland. Its stock markets are therefore used to observe the effects of the developmental stage of the economy for the effectiveness of the trading algorithms.

After the stock markets for this study are chosen, a basket of stocks is gathered from a local index of each country. From the US, we use the New York Stock Exchange (NYSE), Nasdaq Helsinki (OMXH) from Finland and lastly, the National Stock Index of India (NSE) from India. These indices are chosen because they are one of the main indices of each country and the data from them is widely available. The selection of stocks from these indices is made according to the available data from the whole study period, and in this study, we focus on the largest companies of the indices, around 15 to 30 from each one of them. The study period, including the training data required for the model, starts from the year 2010 and ends at the beginning of the year 2024. The data includes historical daily open, closing, high and low prices and also the trading volume of each stock. In total, the dataset therefore includes 14 years of historical data of around 60 stocks from NYSE, OMXH and NSE. The stocks included in these stock baskets can be seen in Appendices 1, 2 and 3. The long study period is selected to mitigate the effects of shocks and temporary trends in the markets and also to observe the effectiveness of the trading algorithm in the long term. In addition, training the neural network model requires a large number of data points in order to increase its accuracy and to avoid overfitting, which might happen with a too small training data set. The larger number of individual stocks is selected for this study for it to be more comprehensive and to provide more context on the consistency of the performance of the trading algorithm. The data for this study was collected from *Yahoo Finance* using Python and its *yfinance* package.

3.2 Methodology

3.2.1 Generation of signals

The trading algorithm used for trading individual stocks requires signals, which it then uses to make investment decisions. These investment decisions are based on the signal's recommendation of a position in the market. The signals are generated using the difference between the current day's closing price and the predicted tomorrow's closing price. Simply put, if the predicted price is higher than the current day's closing price, it indicates that a long position would be optimal to maximize returns. On the other hand, if the predicted price falls below the current price, it indicates that a short position would be able to generate a positive return in the case that the prediction is correct. Despite this being the simplest method to generate signals, in this study, thresholds are used to control the sensitivity of the signal generation. With thresholds, it is possible to control how much the predicted closing price has to deviate from the current price of the stock. In this study, multiple thresholds are used to observe the effect of the sensitivity of the model on the performance to generate returns. The selected thresholds are 0.1%, 0.3%, 0.5%, 0.7% and 1% which means that in the most sensitive case, a 0.1% deviation in the current closing price and the predicted closing price will generate a buy or a sell signal and in the least sensitive case the deviation has to be over 1% for a signal to be generated.

3.2.2 Simulation of algorithmic trading for individual stocks

After the neural network model has been trained for the trading algorithm, it is then used to simulate trading with the testing data. The algorithm iterates through the test period and takes a position in the markets according to the signal from the previous day. All the stocks are simulated individually, and the results are gathered afterwards. The results are used to calculate cumulative logarithmic returns, and the returns are compared to the benchmark strategy of the buy-and-hold strategy. This comparison enables the calculation of excess returns of the trading algorithm across all the stock markets, which in turn helps to answer the question: can the neural network-based trading algorithm generate excess returns while trading individual stocks?

3.2.3 Allocation methods

In addition to trading with individual stocks, trading is also simulated using two allocation methods. These methods differ from individual stock level trading by allocating capital at the portfolio level to those stocks that are predicted to perform better. The same LSTM model and its predicted closing prices are also used for the allocation methods. The allocation methods are rank-based allocation

(RBA) and exponential rank-based allocation (ERBA). They both rely on first ranking the stocks by their predicted returns and then allocating capital based on the ranking. The predicted returns are calculated using the current closing price and the LSTM model's prediction of the closing price of tomorrow. In RBA, 60 % of the capital is allocated to those 20 % of the stocks that have the highest predicted returns and 40 % of the capital is allocated to the middle 30 % of stocks. Therefore, the bottom 50 % of stocks do not get any allocated capital. This method prefers the stocks that are predicted to perform well comparatively. The aggressiveness of the method can be modified by changing the allocation tiers or weights. Compared to the RBA, the ERBA is more aggressive. Similar to RBA, it ranks the stocks based on their predicted returns, but it weighs the upper end of the scale more.

$$W_{i_r} = b^{(N - r_i)}$$

The equation above is used to calculate the raw weight W_{i_r} of the stock *i*. It is calculated based on its rank r_i where *N* is the number of stocks in the portfolio and *b* is the base number which is used to decide on the aggressiveness of the model. In this case, the base *b* is 2, which makes the ERBA more aggressive than the RBA. The raw weights W_{i_r} have to be then normalized to be used in the allocation.

$$W_{i_n} = \frac{W_{i_r}}{\sum_{j=1}^{N} W_{j_r}}$$

The formula above calculates the normalized weights W_{i_n} for the ERBA. It ensures that the sum of weights is equal to 1. The performance of these allocation methods is based on the LSTM's ability to single out the best-performing stocks, even if it wouldn't be able to trade individual stocks effectively. The evaluation of the performance is based on the RBA and ERBA returns, which are compared to an equally weighted portfolio. The equal-weighted portfolio serves as a benchmark for the allocation methods, and it practically generates the average returns of the stocks. If the allocation methods can outperform the equal-weighted method, they can be considered beneficial for investors.

3.2.4 The neural network model

The predictions of closing prices are generated by a neural network model developed specifically for this study. It is developed using Python and its *TensorFlow* package, which is designed for deep-learning applications. The model itself is a Long Short-Term Memory (LSTM) model, a type of Recurrent Neural Network (RNN). The LSTM model is selected as opposed to a vanilla RNN for its improved forecasting accuracy of sequential data, as mentioned in the theoretical framework section of the thesis.

The model operates by first preparing the stock data and separating it into training data and testing data, which is later used for simulation and amounts to 20% of the entire dataset. The model uses the last 60 data points to predict the next day's closing price. The structure of the model includes two LSTM layers with 50 units each and two dense layers with 25 units and 1 unit, the last of which is used to output the predicted price. To handle the weights of units in the model, Adaptive Moment Estimation (ADAM) is used as an optimiser based on the results of the calculated loss function, which in this case is Mean Squared Error (MSE). The model uses 10 epochs, which means that it iterates through the training data 10 times, and the batch size is 32, which means that the model updates its weights every 32 data points. These parameters have been chosen based on their popularity in the development of neural networks and based on the amount of training the model demands before overfitting.

3.3 Evaluation of results

The results are evaluated partly by comparing the simulation results of the trading algorithms to the benchmark strategies. The results are compared across the three chosen stock markets to find out if the returns vary between markets with different features. In addition to these comparisons, this study utilizes the following additional metrics to find out how well the trading algorithm works and also which of the distinct features or factors influence its performance the most. These metrics are used to answer the question of why the trading algorithms works as well as it does.

3.3.1 Accuracy

Accuracy is the simplest form of metric used in the results evaluation. It basically counts how many times the trading algorithm was right and predicted the price movement in the correct direction. Then it displays the accuracy in a percentage calculated with the following formula:

$$Accuracy\ \% = \frac{Number\ of\ correct\ predictions}{Total\ predictions} * 100$$

Accuracy is a simple metric, easy to understand, and it showcases how many profitable trades the trading algorithm can produce if it follows the predictions given by the LSTM model used in this study. It is based on the binarity of market movements, the prediction is either higher or lower than the latest closing price, and the market moves only up and down. The trade is therefore considered to be profitable if the prediction has been made in the same direction as the market moves.

Although easy to understand, the accuracy metric comes with some limitations. It does not consider the magnitude of gains and losses. The algorithm can result in a loss, even though the accuracy is higher than 50 %, if the magnitude of losses is significantly larger than the gains generated by the algorithm. Also, from a risk management point of view, the trading algorithm can be beneficial in cutting losses, even though the accuracy might be under the 50 % threshold. Due to these limitations, accuracy should be only part of the evaluation, not the main metric for it.

3.3.2 Shapley additive explanations

With simple models, the model usually provides the best explanation of the feature's importance (S. Lundberg & Lee, 2017, 2). More complex models, such as deep learning algorithms, require separate explainability models to interpret the original model's functions (S. Lundberg & Lee, 2017, 2). One of these novel explainability models is SHapley Additive exPlanations (SHAP), which is used to assess the importance of each feature of the model. It is one of the most popular methods of determining the importance of features and providing explainability for machine learning models.

SHAP has its foundations in a book written in 1953 by Lloyd S Shapley. The book introduces Shapley values that are used in cooperative game theory. The Shapley value is based on mathematical methods to distribute total gains among the players in the game based on their contributions. In SHAP, players are the features of the original model, and the total gain is the output or prediction of the original model. SHAP therefore uses these Shapley values to assign importance to each feature. The backbone of Shapley values is its tree axioms: efficiency, symmetry and law of aggregation. Efficiency means that the total value of the total gain is distributed fully among the players, and in SHAP, the sum of the Shapley values equals the difference between the baseline values and predicted values of the original model. Symmetry, on the other hand, refers to the equality of contribution and received values of each player. In SHAP, this means that each feature has the same importance as other features with equal contributions. The last axiom, the law of aggregation, means that the sum of values of games equals the sum of values of the players of all games, and in SHAP, this means that the sum of predictions equals to sum of features. (Lloyd S Shapley, 1953, 307–317)

The insights discussed by Shapley are applied in machine learning, most notably in a paper discussing the SHAP framework by Lundberg and Lee (2017). The SHAP utilises the contributions made by Shapley in 1953. Players are the features of the model, the gain is the model's prediction, and Shapley values represent the contributions of each feature of the model (S. Lundberg & Lee, 2017, 4–5). Three key properties of additive feature methods, such as SHAP, are local accuracy, missingness and

consistency (S. Lundberg & Lee, 2017, 4–5). Local accuracy refers to the equality between the explanation model and the original model.

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x_i'$$

This equality is shown in the formula above. $f(x)$ is the original model and $g(x')$ is the explanation model and their values should be equal based on local accuracy. The *x* is the original input of the model, and the x' is the input of the explanation model. On the right side of the formula, ϕ_i refers to the effect of each feature, and *M* is the number of features. This formula has the same implications as the efficiency in Shapley values, as it implies that the sum of feature attributions' significance equals the difference between the baseline and predicted values. The second property of additive feature methods is missingness. It means that if the feature does not contribute to the predicted value, it has a feature significance value of zero.

$$x_i' = 0 \Longrightarrow \phi_i = 0$$

The equation above shows that if the feature's input x_i' is zero, then the impact ϕ_i is also zero. The property, consistency, means that if the model changes and the contribution of some input increases, the feature importance should not decrease.

$$f_x'(z') - f_x'(z'\backslash i) \geq f_x(z') - f_x(z'\backslash i), \qquad z' \in \{0,1\}^M$$

The consistency property is defined with the formula above. z' is a binary input referring to the feature either present (1) or absent (0). $z'\backslash i$ is the input set to zero. The difference between $f_x(z')$ and $f_x(z'\backslash i)$ is a marginal contribution of feature *i* to the model's output. The formula implies the following:

$$\phi_i(f', x) \geq \phi_i(f, x)$$

These formulas can be explained in the following manner: if the marginal contribution of feature *i* of the updated model f' increases compared to the model *f*, the SHAP value of ϕ_i for feature will also increase. A formula for calculating the SHAP value that satisfies all the properties above is below.

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!\,(M - |z'| - 1)!}{M!} [f_x(z') - f_x(z'\backslash i)]$$

In the SHAP value formula, $|z'|$ is the number of values in z' that are not zero and $z' \subseteq x'$ refers to vectors z' that are a subset of vectors x'. Using the values from the function $\phi_i(f, x)$, one can calculate the output *f(x)* of the model. (S. Lundberg & Lee, 2017, 4)

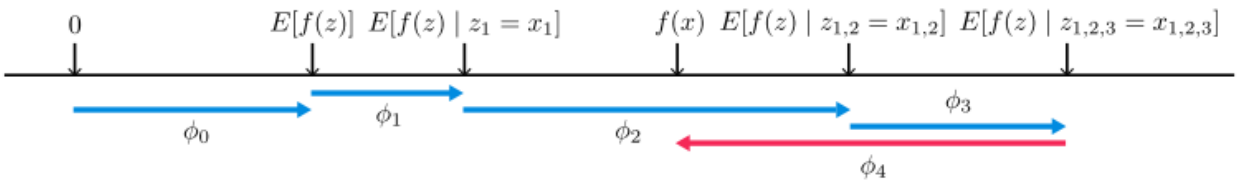

Figure 6 Deconstruction of model output into SHAP values (Lundberg & Lee, 2017)

The figure above visualises the additive function of SHAP where the sum of SHAP values equals the output of the predictive model. This additive function of SHAP makes it intuitive to understand the weight of each variable in complex models, which are used, e.g. in machine learning.

SHAP is a widely used tool in finance applications. Unexplainable black box models cannot be used in financial services, which makes models like SHAP necessary (Bussmann et al., 2021, 203–204). Explainability is, therefore, a necessity in all kinds of modelling in finance applications. It refers to the ability to understand the decision drivers of the model, which can be presented to stakeholders (Bussmann et al., 2021, 204). In addition to this, explainability can become a legal issue as it can be mandatory at some level to provide explainability, e.g. according to the European Union general data protection regulation (GDPR, 2016/679), the subject of the data can be entitled to information responsible for the decision of an automated decision making. Under these circumstances, the use of complex artificial intelligence or machine learning models requires some form of explainability in financial applications. The wide range of use cases of SHAP in finance provides backing for its usefulness in explaining the decision-making drivers of the models.

One of the main strengths of SHAP, which makes it so popular, is its ability to be used in any kind of machine learning model. SHAP is therefore model-agnostic and is used in models such as linear regression, decision trees and deep learning networks (S. Lundberg & Lee, 2017, 5–6). This is a great benefit of the model as it enables its general use without the need for users to research model-specific explainability models or alterations for model-specific use cases. Another property that SHAP has that increases its general use is its global and local interpretability. SHAP provides explanations globally in general for the entire model and also locally for individual predictions (S. M. Lundberg et al., 2019, 1). It can therefore be used to provide insights into each individual prediction it outputs which can be beneficial in exploring reasons behind outliers or errors in predictions and also to

provide information about the importance of each variable for the whole model. This information can be used to improve the accuracy and efficiency of the model and its predictions, leading to more accurate results. SHAP is also compliant with EU GDPR about the explainability of AI models (Misheva et al., 2021, 2). Used AI models cannot be just considered black boxes, but the user of an AI model must be able to explain the model to 3rd parties. This regulation applies, for example, within financial services that utilise AI and machine learning models in fraud detection, risk management and other functions. Lastly, one important property of SHAP is its consistency. If a model changes such that some variable becomes more important, SHAP will reflect this change and increase the weight of it in the results (S. Lundberg & Lee, 2017, 4). SHAP can therefore be used to monitor changes in the model or the data and its dependencies between the variables and the output. This can be beneficial when the user of an AI model is developing or trying to improve the accuracy of the model, or when the user might suspect incorrect predictions and want to confirm if the model is working as it should.

The mentioned strengths of SHAP have made it popular among implementers of AI and machine learning models, and why it is chosen as one of the methods to provide explainability of the used model in this study. SHAP will be used to interpret the importance of the features in the data that is used to generate predictions of future price changes. The values are calculated with relational values and also absolute values. The relational value shows the direction and weight of each variable and its effect on the prediction, and the absolute value shows the absolute impact, which is also important, as in some cases, the variable impacts the predicted value positively or negatively at different times, meaning that the relational value can be minimal if the impact is distributed evenly between the positive and negative effects while the average of the absolute values shows a significant impact.

4 Results

The LSTM-based trading algorithm is first assessed by its ability to generate excess returns in the three stock markets, which are the NYSE, OMXH, and NSE. The analysis includes the observation of excess returns of each individual stock and their average, followed by the analysis of the distribution of the total logarithmic return of each stock. These results are then compared across each stock market in order to find out if the performance of the trading algorithm depends on the stock market with different characteristics. The excess return analysis uses the signal sensitivity threshold of 0.005, implying that the difference between the current price and the predicted price has to be higher than 0.5% for the signal to be generated for the next day. In the next step, the impact of different signal sensitivities is analysed by both increasing and decreasing the signal sensitivity and analysing the results generated with those signal sensitivity thresholds. Then the two allocation methods, RBA and ERBA, are analysed for their ability to outperform the EWP in each of the stock markets. Transaction costs are considered for those strategies that are able to outperform the benchmark to assess their ability to be used in the real world more thoroughly. Finally, the model and its predictions are evaluated using the accuracy and SHAP to observe the feature importance and the accuracy of the model. Based on this comprehensive analysis, we will be more informed of the performance and the ability of the model in algorithmic trading.

4.1 Excess returns

4.1.1 NYSE

The first set of stocks is analysed from the New York Stock Exchange (NYSE). The outcome of this analysis serves as a benchmark for later analysis of the results of trading on other stock markets. The cumulative logarithmic excess returns can be observed in the figure below.

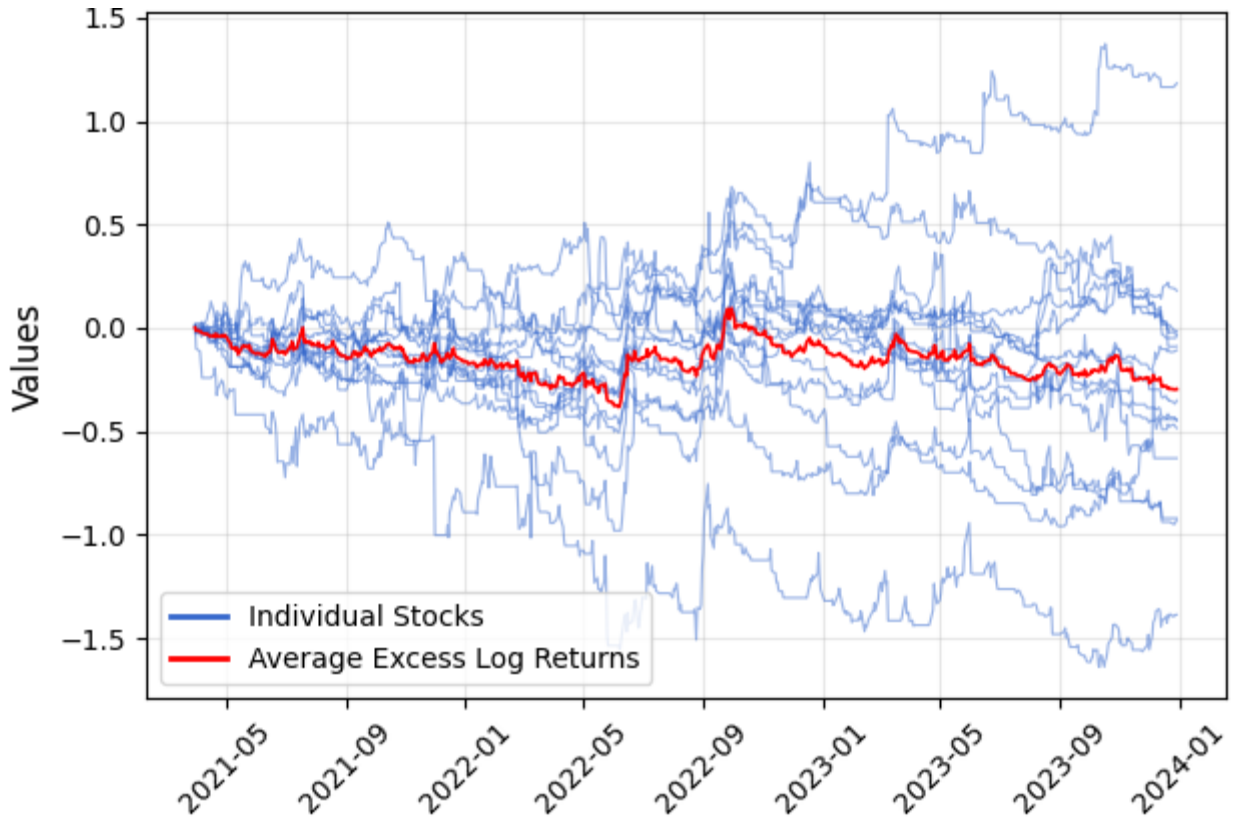

Figure 7 NYSE cumulative logarithmic excess returns

In this figure, we can see the cumulative logarithmic returns of individual stocks, which are visualised with blue lines and their average is shown with the red line. Initially, it is clear that there is a wide range of outcomes in the results of excess returns from the LSTM-based trading algorithm compared to the buy-and-hold method used as a benchmark. The y-axis signifies the excess returns using the multiplier that indicates the amount that the return of the trading algorithm differs from the benchmark returns shown in logarithmic returns. Now, if we observe the total cumulative logarithmic excess returns on the individual level, the highest value is about 1.2 while the lowest value is around -1.4, indicating that the trading algorithm's performance varies based on the stock and the data it provides. Therefore, it is not possible to conclude that the trading algorithm will always outperform or lose to the benchmark method. Due to this, we have to look at the general results, which means here that we look at the average of the excess returns observed. The average line has its ups and downs during the observed time frame, and despite that the total excess returns are negative, there is a short time period that which the value is positive, at the end of 2022. At that time, there is a clear trend of increased excess returns with almost all of the individual stocks. A similar trend can be observed earlier during the year 2022, where the average excess returns increase strongly due to most individual stocks generating excess returns at that time. This reappearing trend would suggest that there could be an

underlying trend in the markets that the trading algorithm is able to detect and utilise to outperform the benchmark strategy at that time. To confirm this, we have to compare the returns to the market index.

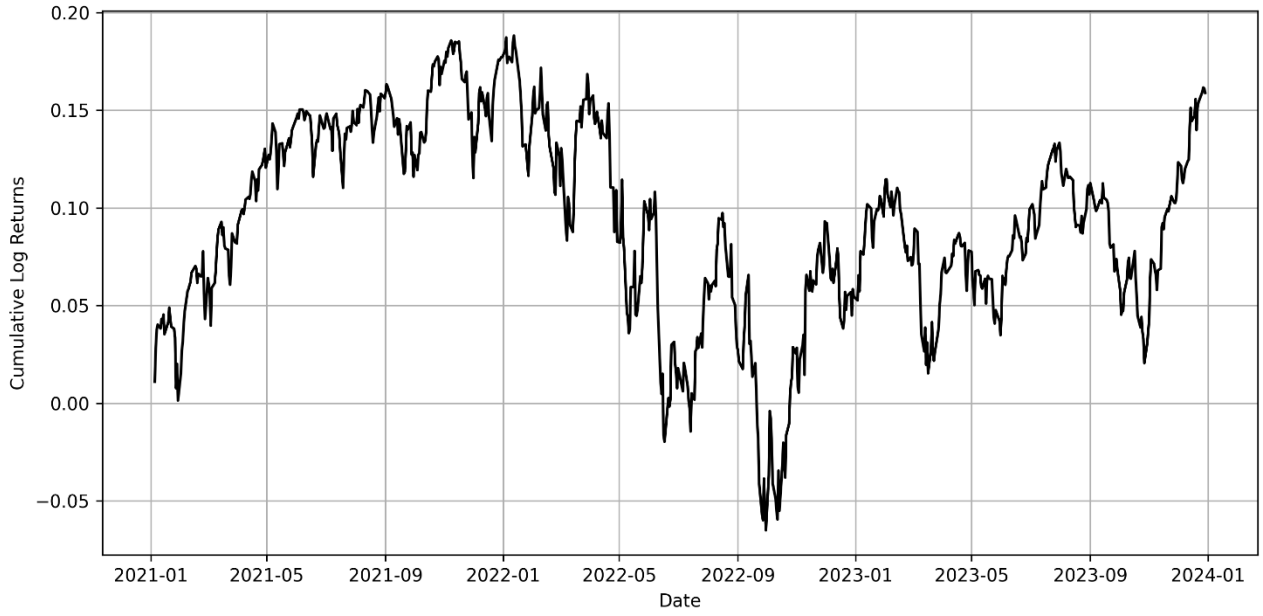

Figure 8 NYSE index time series

Above is a visualisation of the NYSE index time series in cumulative logarithmic returns. Using this chart and by comparing it to the returns of the NYSE index stocks, we can observe that the sudden increases in excess returns are around the same period as when there are significant decreases in the index. Those decreases are around the summer of 2022 and October of 2022, which are both the same periods that had jumps in the excess returns. This would indicate that the trading algorithm can detect these sudden downturns in the data and generate preferable signals based on this. Outside of these well-performing time frames, the trading algorithm seems to lose to the benchmark on average. This seems to be especially true during strong upward trends, indicating that the trading algorithm is not utilising the entire strength of the trend for its benefit. Due to the index as a whole having an upward trend most of the time compared to a couple of sudden drops in the index, the total excess returns are negative on average, which can also be observed from the next figure.

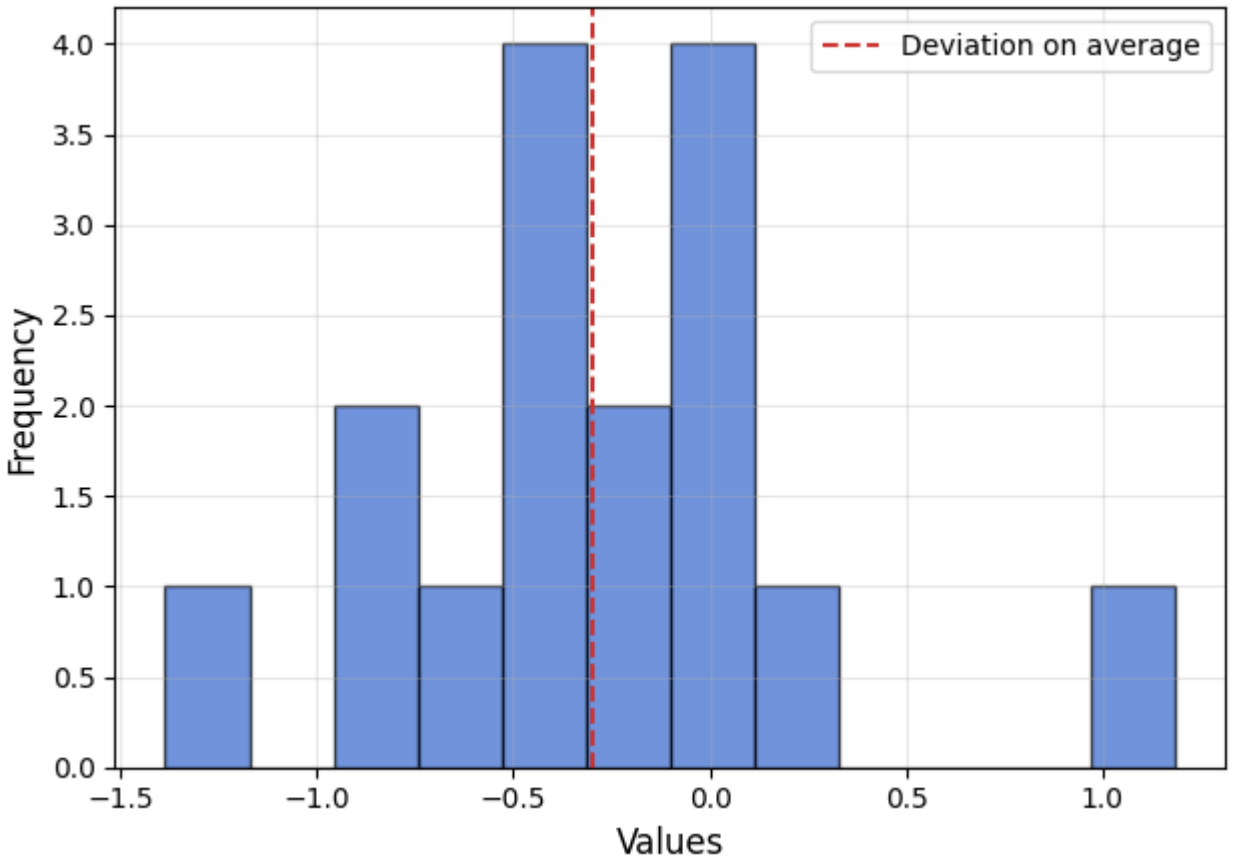

Figure 9 NYSE total logarithmic excess returns distribution

The histogram above visualises the total logarithmic excess returns that the individual stocks had. The frequency of stocks that have a certain total excess return value is shown with the blue bars, and the average of these returns is shown with the red line. By analysing the histogram, it is obvious that in most of the occurrences, the trading algorithm was not able to beat the benchmark strategy of buy and hold.

4.1.2 OMXH

Next, we will analyse the trading algorithm's results on stocks from the Finnish stock market, Nasdaq Helsinki (OMXH). Compared to the NYSE, OMXH's main characteristics are that the combined value of the firms and the number of transactions and investors on OMXH are significantly lower than on the NYSE. Intuitively, this could benefit the trading algorithm due to less competition, but that also requires the trading algorithm to spot patterns and generate accurate signals based on those.

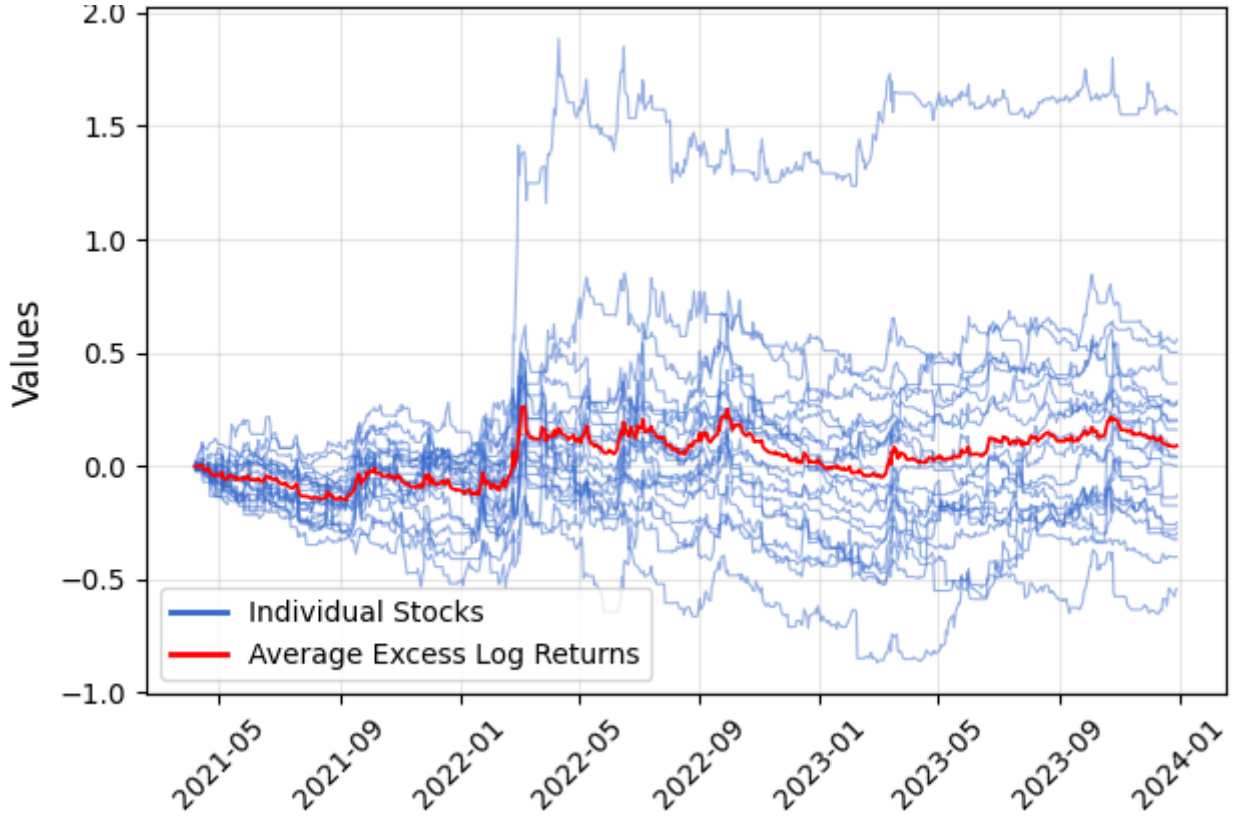

Figure 10 OMXH cumulative logarithmic excess returns

The results of the trading algorithm and its excess returns can be seen above, and the figure can be interpreted similarly to the same figure in the NYSE section. Compared to the results on NYSE, the excess returns on OMXH seem to be more positive. By observing the time series individually and also on average, there are multiple upward spikes in excess returns and in addition, there is an increasing trend in excess returns from the first half of the year 2023 to the end of the year 2023. The most significant spike in excess returns can be observed during the first quarter of the year 2022. The spike increases the value as much as 0.5, which brings the excess return level from negative to positive. In addition to this large spike, similar, although less impactful spikes can be seen during the last quarter of 2021m the second quarter of 2022 and the second quarter of 2023. These spikes are not as impactful, but in combination, they increase the total result significantly. These periods of increasing excess returns have the combined impact of increasing the excess returns on a positive level. As with the analysis of excess returns on the NYSE, these distinct events and periods are compared to the OMXH stock market index to determine how the performance of the trading algorithm depends on the market conditions. The OMXH index time series can be seen below.

Figure 11 OMXH index time series

The figure above shows the OMXH index's time series in cumulative logarithmic returns, which can be used to observe significant price movements in the markets. These include the strong upward trend during the year 2021, which follows a sudden drop in the index at the start of 2022. The rest of the time series chart includes a general downward trend with some up-and-down movements. Now, when the significant price movements are compared to the excess returns from the figure before, we can see that the spikes happen around the same time as the significant drops in the index, and in addition, the trends in the index seem to cause a contrary trend in the excess return chart. This would indicate that the trading algorithm results have a similar dependence on the index as with the NYSE index, as the market returns are dampened by the trading algorithm, resulting in decreased returns and losses compared to the benchmark strategy. In the case of trading with stocks from OMXH, the results are more positive compared to the NYSE stock returns, which most likely is a result of a general trend in a different direction. The NYSE index trend was generally increasing as while OMXH, the general trend is decreasing. This would seem to be a large factor in why the excess returns are higher with OMXH compared to NYSE.

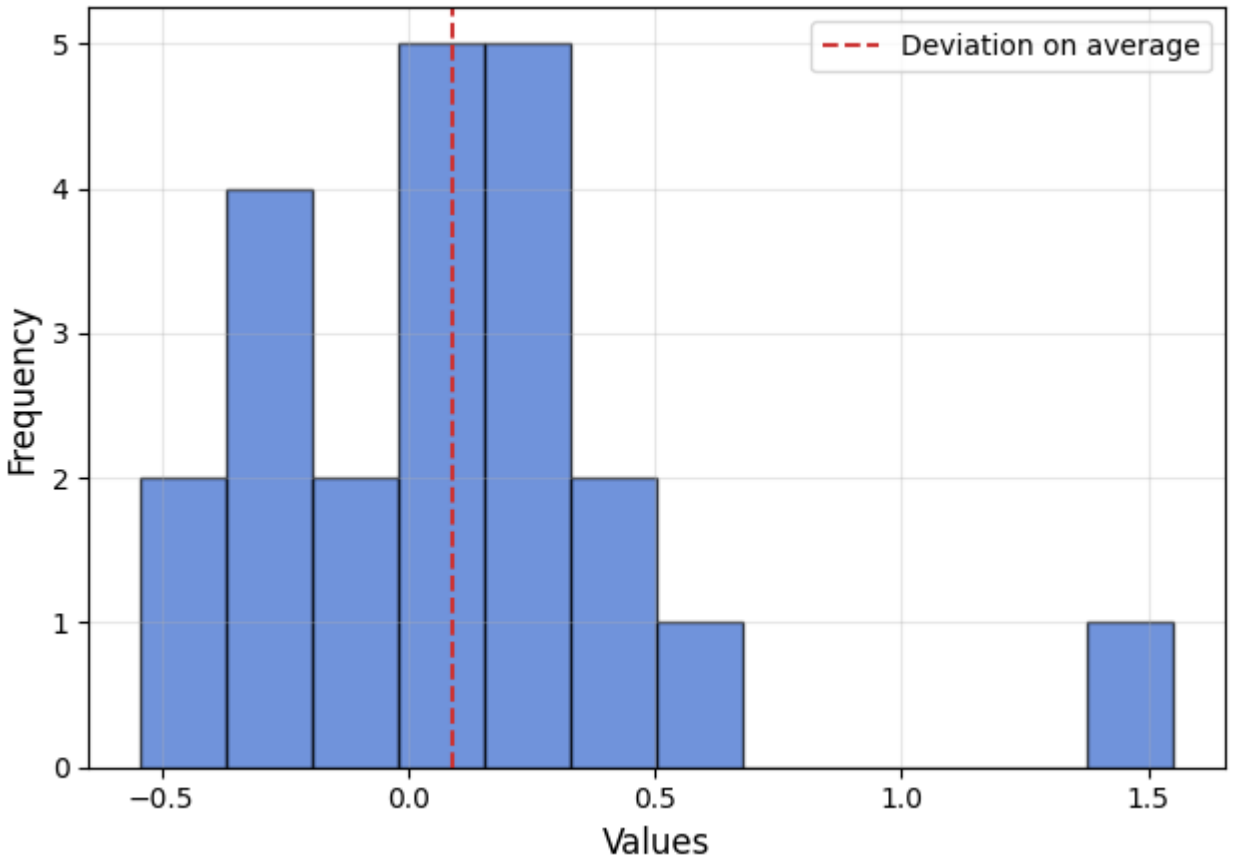

Figure 12 OMXH total logarithmic excess returns distribution

Now, if we analyse the figure above, which illustrates the outcome of the entire trading period for each stock, we can observe that most of the total logarithmic excess returns for individual stocks are between -0.5 to 0.5, while the average is barely positive. This means that generally, the trading algorithm is not a reliable method to outperform the buy-and-hold method, but on average, it was able to do this. In one instance, the excess return is around 1.5, which is an outlier.

4.1.3 NSE

Lastly, we will look at the results of trading in the Indian stock market National Stock Exchange of India (NSE). Compared to Finland and the USA, India is considered less economically developed, but its economy is substantially larger than the Finnish market. NSE serves as an example of what happens with the results of the trading algorithm when the developmental level of the country is used as a variable and the results are compared to the benchmark market of NYSE.

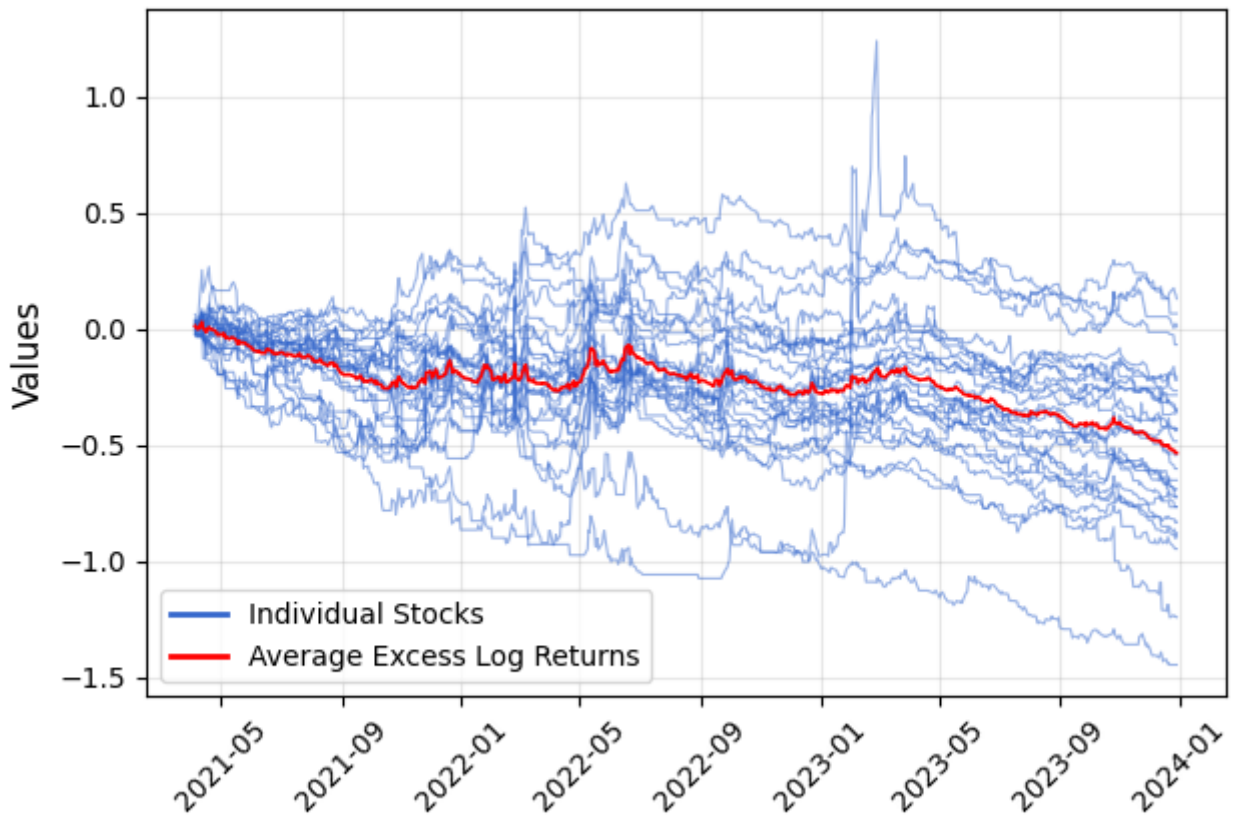

Figure 13 NSE cumulative logarithmic excess returns

Above is a chart of the cumulative excess returns of the trading algorithm compared to the buy-and-hold method in the NSE. The price path of stocks has some variance, but the general trend seems to indicate that over time, the trading algorithm is not able to outperform the buy-and-hold method. This observation is in line with the results from NYSE and OMXH. The downward trend of excess returns is even clearer with NSE than with the earlier results. The observation period starts with a downward trend until the end of 2021 which is followed by a more volatile period of excess returns with some signs of positive results until the second half of 2022 when the average line stabilizes and the downward trend continues until the end of the observation period, the start of the year 2024. With the previous NYSE and OMXH analysis, changes in the excess returns could be at least partly explained by comparing the excess return chart to the stock index chart of each stock market. A similar analysis can also be done with NSE with the chart below.

Figure 14 NSE index time series

The figure above shows the NSE stock index from 2021 to 2024 in cumulative logarithmic returns. The general trend of the index is upwards, with some increased volatility from the end of the year 2021 to the second quarter of 2023. These trends coincide with the average excess returns on the previous chart. During the upward trend of the index, the excess returns decrease, and during the volatile period, the excess returns are also volatile. This inverse dependence of excess returns and stock index performance is in line with the NYSE and OMXH results. This suggests that, similarly to those stock markets, the trading algorithm is not able to utilise the full effect of the increase in the price of stocks and therefore performs worse than the benchmark strategy of buy and hold during the bull markets.

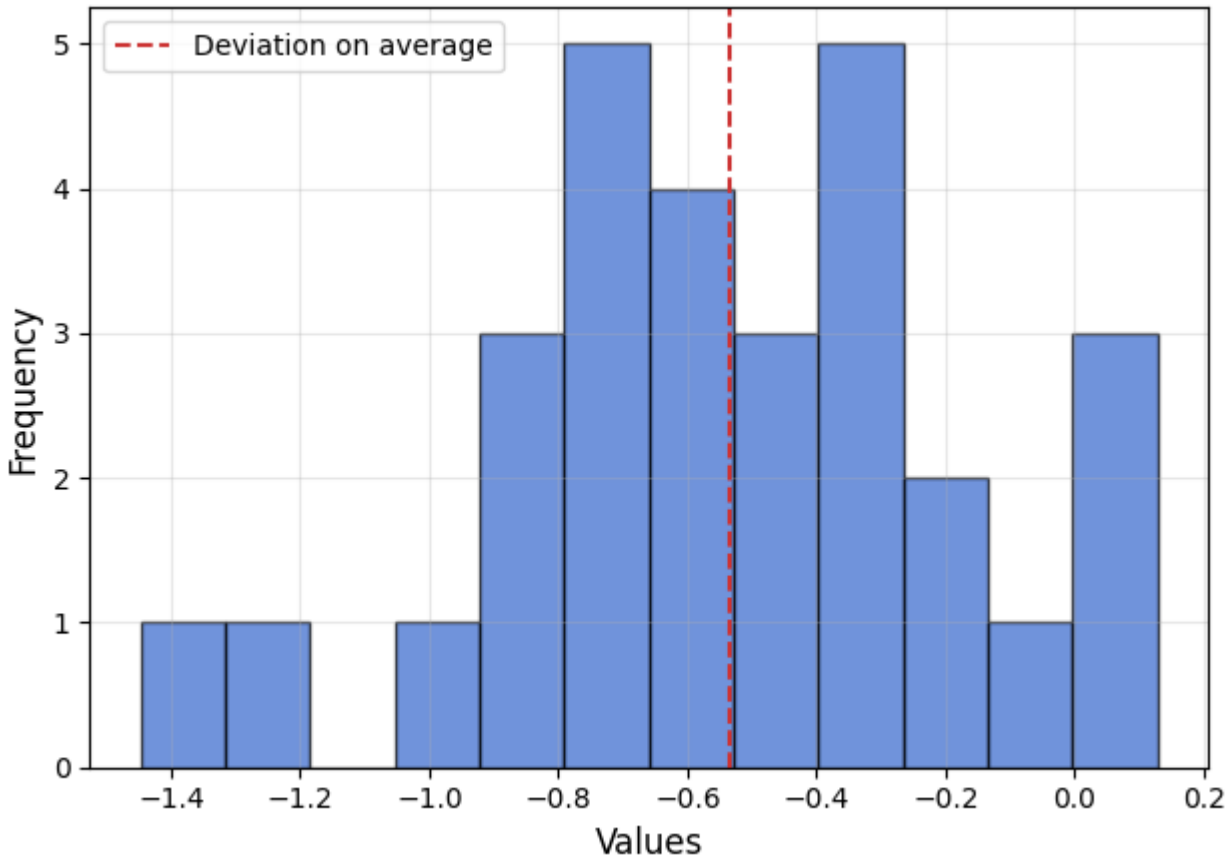

Figure 15 NSE total logarithmic excess returns distribution

Finally, the histogram above showcases the final values of the excess returns of the trading algorithm compared to the benchmark buy-and-hold strategy. The results are the most obvious out of all of the markets in this study. The trading strategy performed worse than the benchmark with all but 3 stocks, which indicates that the trading strategy cannot outperform the buy-and-hold strategy in the NSE. Based on the results seen from each market, it seems that the trading strategy dampens the effect of trends in the markets by not participating in the markets as frequently as the buy-and-hold strategy does. This means that while the price moves in a direction, the trading strategy holds a long position for only some of the observation period, whereas the buy-and-hold strategy continues to benefit from rising trends. NSE has the strongest upward trend out of the observed stock markets, which, combined with the dampening effect, leads to the negative results seen above.

4.2 Signal sensitivity

The trading algorithm makes its investment decision based on signals produced by the LSTM model's predictions of the closing prices of individual stocks. These signals are set to be generated at a certain threshold of sensitivity. The base sensitivity of the signal is set at 0.05 %, meaning that the predicted

closing price has to deviate at least 0.05 % of the latest closing price for the signal to be generated. By varying the signal sensitivity, we can observe how the sensitivity of the signal affects the outcome of the trading algorithm. The alternative signal sensitivities in addition to the 0.05 % are 0.001 %, 0.03 %, 0.07 % and 0.1%.

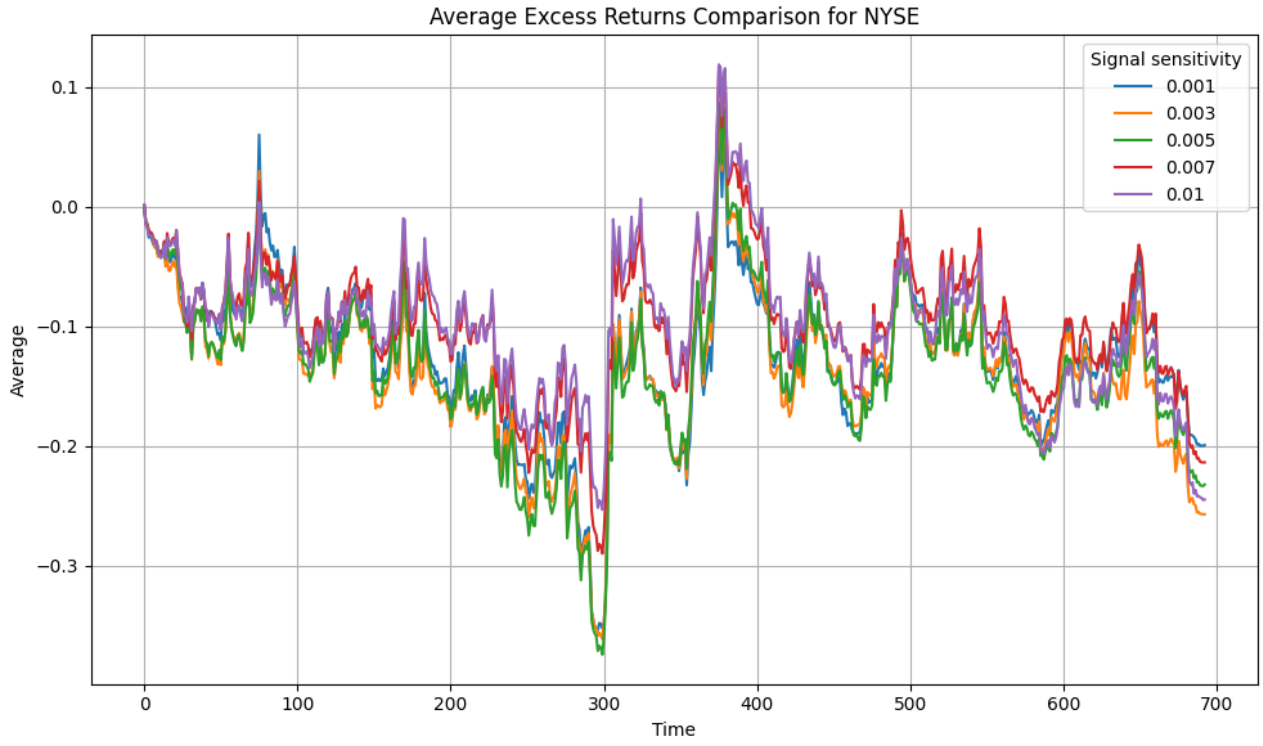

Figure 16 NYSE signal sensitivity comparison

In the above chart, we can observe the effect of different signal sensitivities on average logarithmic excess returns for the NYSE. All of the price paths seem to follow similar trends with some limited deviations. The total cumulative excess returns at the end are also very similar, however, the sensitivity of 0.001 %, which is the blue line, appears to be the highest. When examining the entire observation period, this is not the case, and the best-performing strategy changes from time to time. Generally, the sensitivities of 0.07 % and 0.1 % are highest out of the lines, at least most of the time, which would indicate that the weaker sensitivity could be beneficial for the trading algorithm. Despite this, all of the excess returns on different sensitivities are negative, suggesting that by changing the sensitivity, it is not possible to make the algorithm outperform the benchmark on the NYSE. Additionally, the minimal deviation, despite the varying signal sensitivity, suggests that the sensitivity is not a very impactful factor, at least on the scale used and the stock market.

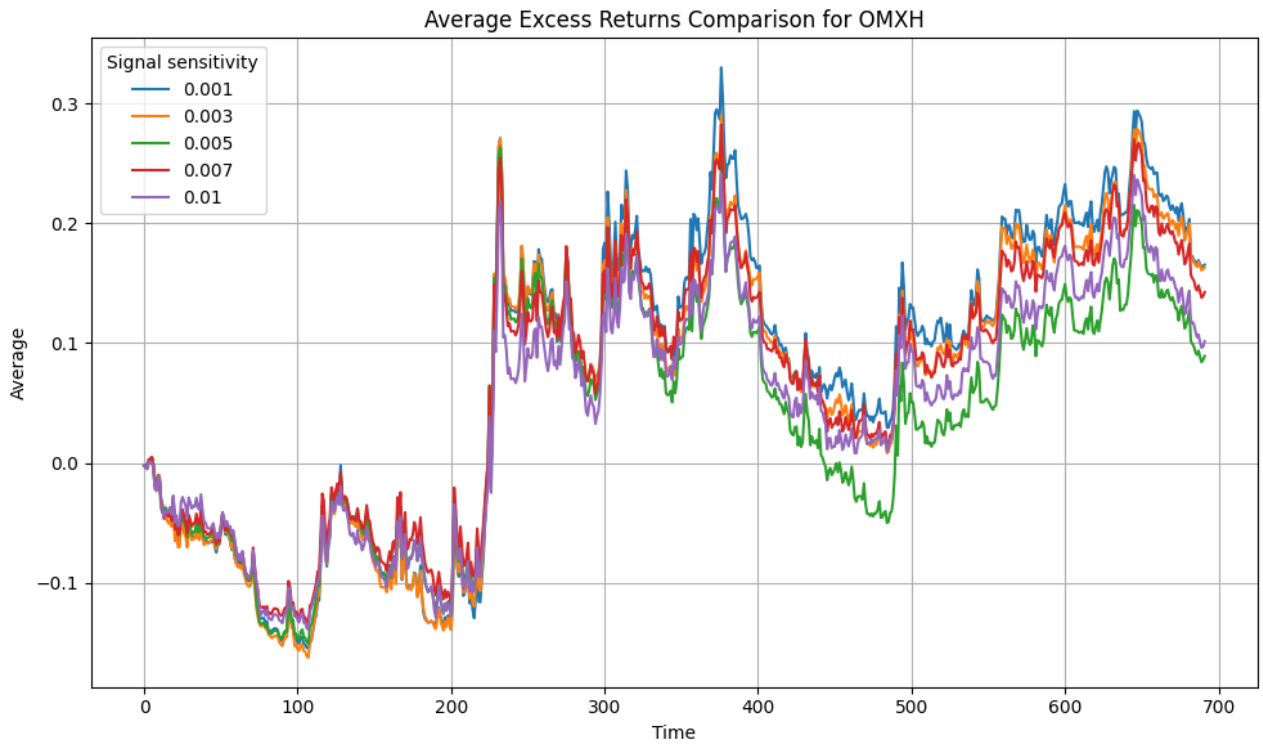

Figure 17 OMXH signal sensitivity comparison

The figure above presents the same information as the previous chart, but with data from OMXH stocks. Similarly, the trend between the lines indicating different sensitivities appears to be quite close together. The largest deviation is observed after the halfway point of the time series, with the green line indicating a sensitivity of 0.05 % as it falls below the other lines. This leads to the lowest excess returns for that sensitivity throughout the entire period following the event, resulting in the total excess returns at the end being the lowest for that strategy. Conversely, at the other end of the scale, the higher-performing strategies with sensitivities of 0.01 % and 0.03 % yield the highest total cumulative logarithmic returns. This differs from the earlier observation that lower sensitivity would be advantageous for the trading strategy since stocks from OMXH seem to generate higher returns when sensitivity is increased. However, a distinct feature separates the OMXH and NYSE stock markets: during the observation period, the NYSE's index is increasing, while the OMXH's is decreasing. Although the difference in the strategies is minimal, it could influence the outcome.

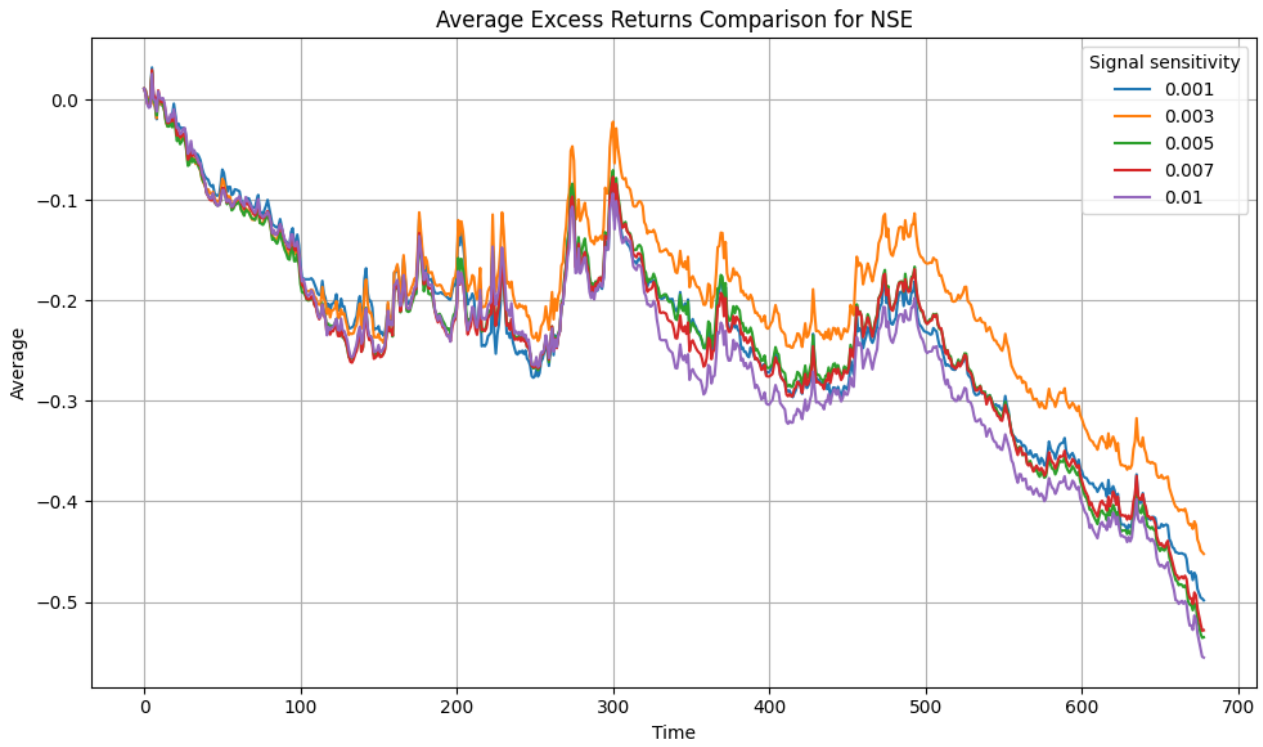

Figure 18 NSE signal sensitivity comparison

Lastly, the figure above showcases the average excess cumulative returns of the trading strategies with different signal sensitivities on the NSE market. Similar to earlier results, returns with all sensitivities follow similar trends, and all of the total excess returns are negative. Despite this, there is a somewhat clear outlier, the 0.3 % signal sensitivity, which produces the highest results. The lowest results are produced with the 1 % signal sensitivity. This indicates that the lower sensitivity thresholds perform slightly better than the higher sensitivity thresholds, which is in line with the OMXH results and contradicts the NYSE results.

All things considered, the trading strategy is not able to produce excess returns on the NYSE or the NSE, but on OMXH, it produces positive results. The suspected reason might lie behind the general trends of the indices, which is also discussed in the “Excess results” part of the thesis. The NYSE and NSE are increasing during the observation period, and the OMXH is decreasing during the period. The changes in the excess returns occur simultaneously with the trends in the indices in the inverse direction. This suggests that the trading strategy is merely dampening the effect of the trends on the market, and now, after the analysis of the sensitivities, it strategy can’t be significantly improved by varying the signal sensitivity.

4.3 Allocation methods

The other approach of utilising the predictions produced by the LSTM model is to allocate the capital to those stocks that are predicted to perform better, instead of just trading with individual stocks. The Allocation methods used are the rank-based allocation (RBA) and exponential rank-based allocation (ERBA), which are both based on ranking the stocks based on the predicted returns and then allocating the capital to those stocks. The results are compared to the benchmark strategy of an equally weighted portfolio (EWP).

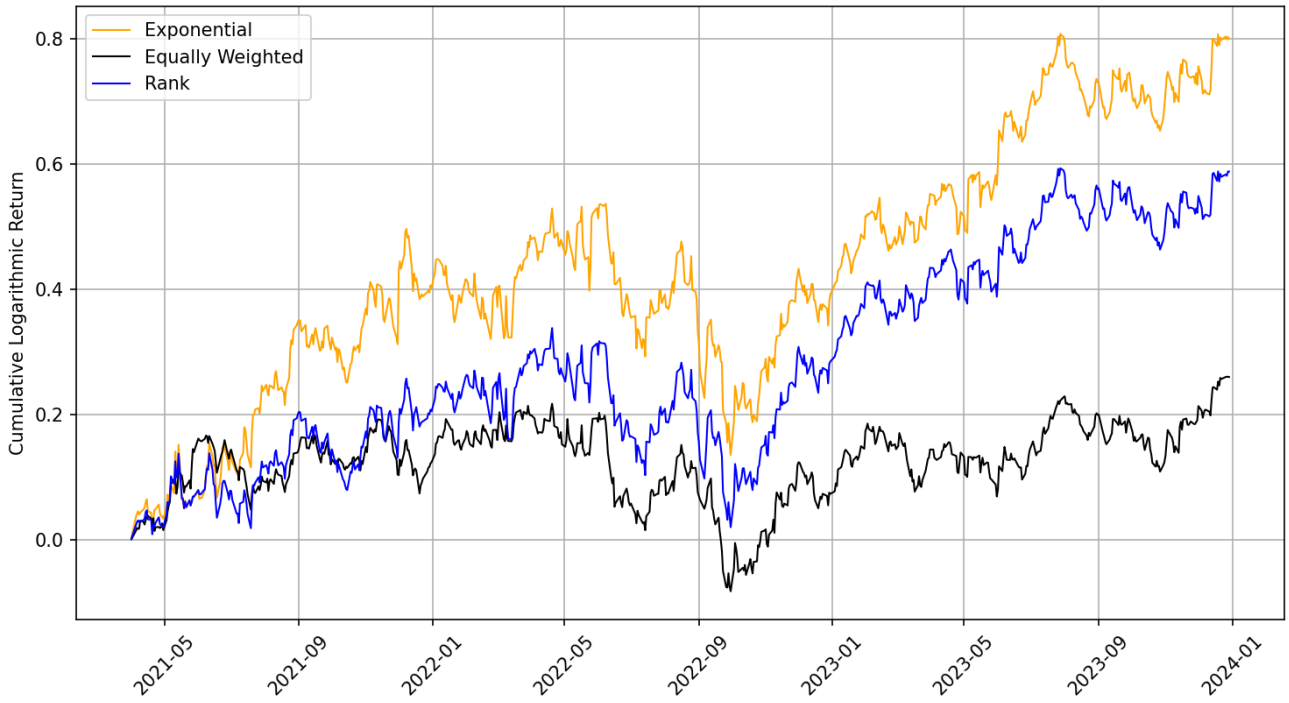

Figure 19 NYSE cumulative logarithmic returns for different allocation strategies

Above is a line chart illustrating the logarithmic cumulative returns of various allocation methods in the NYSE stock market. The yellow line represents the ERBA, the blue line indicates the RBA, and the black line depicts the EWP. Based on the chart, the ERBA is the best overall performer, generating approximately 80 % of logarithmic returns by the end of 2023. It demonstrates a strong capacity to recover after losses. Notably, these drawdowns appear to impact it more significantly than the other strategies. In addition to drawdowns, ERBA seems to amplify all trends shown in the chart, making it seem somewhat more volatile than the other strategies. In comparison to the ERBA, the RBA is somewhat more stable, yet it still outperforms the EWP, generating nearly 60 % of logarithmic

returns. The allocation strategy is less aggressive than that of the ERBA, yet it remains almost as volatile. The EWP reflects market trends and performance without taking a stance on individual stocks and their future outcomes. The EWP generates about 25 % of logarithmic returns, with the RBA and ERBA significantly outperforming it. Consequently, the LSTM model appears to be suitable for ranking stocks and predicting their future performance on the NYSE.

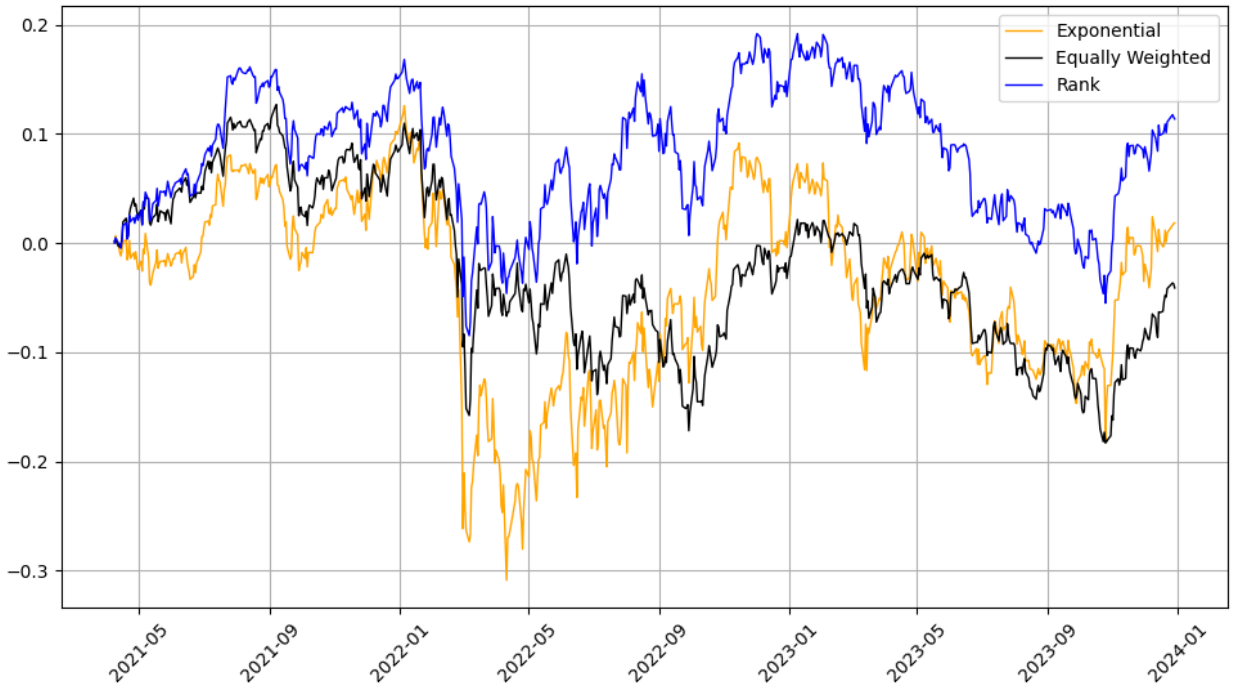

Figure 20 OMXH cumulative logarithmic returns for different allocation strategies

The figure above shows the cumulative exponential logarithmic returns of the three strategies on the OMXH market. The best performer this time is the RBA, which generates just over 10 % in logarithmic returns. Its returns consistently exceed the EWP throughout the entire observation period. In contrast, ERBA has periods where its cumulative logarithmic returns fall significantly below the EWP. However, by the end of 2023, its total returns surpass the EWP and remain positive. A notable negative period for ERBA occurs in the first half of 2022, during which its higher volatility results in a substantial decline in returns, even compared to the other strategies, which also face significant losses. This illustrates the downside of aggressive and risky allocation. Despite this sudden downturn, even ERBA is able to outperform the EWP eventually. These results are in line with the results from NYSE, supporting the fact that the LSTM model is able to create accurate rankings of which stocks will perform comparatively better than the EWP.

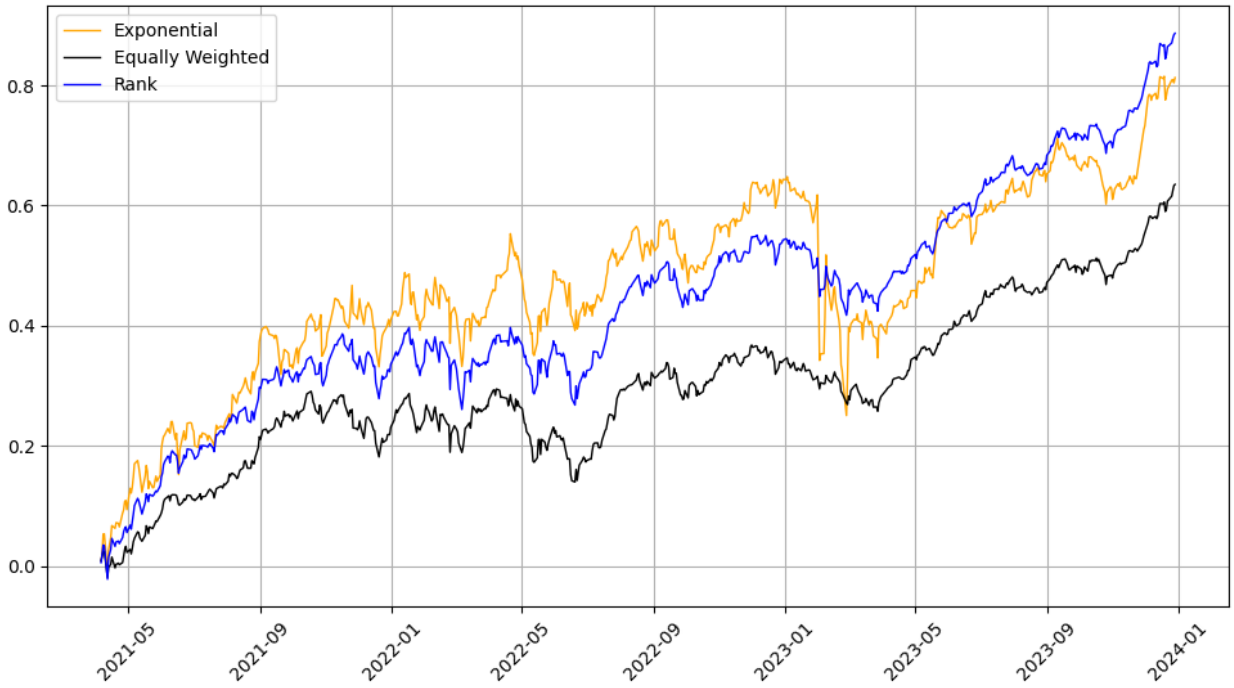

Figure 21 NSE cumulative logarithmic returns for different allocation strategies

Lastly, above are the results of the trading strategies on the NSE stock index. From the beginning of the observation period up until the start of the year 2023, the ERBA is generating the highest returns and outperforming the other strategies. This eventually changes as the ERBA experiences a sudden downfall, limiting its cumulative logarithmic returns almost to the same level as EQP. After this event, the RBA stays on top until the end of the observation period with its above 80 % of logarithmic returns. Despite the sudden downfall, the ERBA is almost able to catch up to RBA, and its total cumulative logarithmic returns are around 80 %. The returns of these strategies are the highest on NSE compared to the previous markets, but so are the returns of the EWP. It generates over 60 % of returns, which are the highest out of the observed markets. The sudden downfall of the ERBA is again due to its aggressive and risky allocation. The strategy allocated 50 % of its capital during the downfall to a stock Adanient, which lost around 54 % of its value in only six trading days. This is again an example of the drawbacks of the aggressive allocation. Excluding this one event, the returns in general are the most consistent across the strategies when compared to the other markets. The benefit of RBA and ERBA are therefore a bit dampened, especially when compared to NYSE, but they still outperform the benchmark of EWP, indicating the effectiveness and accuracy of the LSTM model in ranking the stocks based on their predicted performance.

In conclusion, the RBA and ERBA are able to outperform the benchmark strategy of EWP in all three markets. This supports the predictive power of LSTM across different market sizes and economic

development levels. In the NYSE, ERBA was clearly the best performer, and it seems to benefit from the upward trend of the markets. In contrast to this, OMXH has more varying trends, with EWP ending up with negative returns. This probably made it harder for the LSTM model to predict a couple of clear winners and resulted in worse results for ERBA. RBA, on the other hand, performed better and despite the trend pointing generally downwards, it was able to turn in profits, outperforming the two other strategies. In addition, ERBA experienced the risks of increased volatility, resulting in lower returns than RBA. The same can be said about the NSE, where ERBA seems to first outperform the other two strategies, but due to aggressive allocation, it experiences a strong drawdown. These results indicate that the RBA is more stable and more reliable at outperforming the markets, but the potential for higher returns with ERBA can also be observed.

4.4 Transaction costs

After establishing the profitability of the trading algorithm without considering transaction costs, this section examines how transaction costs impact profitability and viability of the trading strategies. As previously concluded, trading on individual stock level underperformed compared to its benchmark strategy buy-and-hold, which is why transaction costs are not considered with individual level trading. Having said that, it should be noted that including transaction costs would have further deteriorated those results. In contrast, portfolio level allocation strategies were able to outperform its benchmark EWP, which makes it interesting to look into the impact of transaction costs and with what cost level, the allocation methods are able to outperform the benchmark. The analysis is done by simulating the trading with different levels of transaction costs for each change of allocation. The allocation levels are 0.005%, 0.01% and 0.015% per trade. EWP is used as a benchmark for comparison and assumed to be cost-free due to its passive nature.

	No Costs	**0.005%**	**0.01%**	**0.015%**	**EWP Return**
Rank Based Allocation					
OMXH	11 %	-5 %	-21 %	-38 %	-4 %
NSE	89 %	76 %	63 %	50 %	64 %
NYSE	59 %	39 %	19 %	-1 %	26 %
Exponential Rank Based Allocation					
OMXH	2 %	-21 %	-43 %	-66 %	-4 %
NSE	81 %	61 %	41 %	21 %	64 %
NYSE	80 %	57 %	33 %	10 %	26 %

Table 1 Total logarithmic returns of allocation methods with transaction costs

The table above showcases the total cumulative logarithmic returns for each market and cost scenario for both of the allocation methods. With both methods in all of the markets the results without transaction costs are higher than with the EWP. Now by incorporating transaction costs into the trading, we can observe a significant impact on the final results. With 0.005% cost level RBA is still able to outperform EWP on NSE and NYSE but with 0.01% cost level, the results fall slightly below the EWP. This indicates that RBA is only effective on these markets if the transaction costs are below 0.01% and within OMXH, the cost level should be below 0.005% for RBA to outperform the EWP. With ERBA, a similar but a more sensitive cost pattern is observed. At 0.005% cost level, ERBA already falls below the EWP in OMXH and NSE. With NYSE, ERBA is able to outperform the EWP even with 0.01% cost level and only at 0.015% we can see that it falls below it. Notably, ERBA seems to be more sensitive to transaction costs than RBA which is likely due to its more aggressive allocation mechanism, increasing its transaction sizes. In conclusion, active allocation methods require transaction costs below 0.005%-0.01% to outperform the passive benchmark.

4.5 Evaluation and feature importance

While analysing the returns of the LSTM-based trading algorithm provides insights into the model's usefulness, it is equally important to examine the model itself to understand the inner workings of the trading algorithm and the factors that lead to its results. This chapter evaluates the accuracy of the model's forecasts and the significance of the input variables, also known as features. The accuracy analysis compares the predicted price movements generated by the LSTM network to the actual price movements of the stock at each time point.

	Signal sensitivity				
	0.1 %	0.3 %	0.5 %	0.7 %	1 %
NSE	0.497	0.495	0.480	0.478	0.449
NYSE	0.486	0.483	0.480	0.443	0.418
OMXH	0.490	0.487	0.481	0.478	0.480

Table 2 Accuracy metrics of LSTM forecasts

Above is a table that shows the accuracy of the LSTM model's predictions compared to the actual price movements. Each row shows the average results for stocks in a specific stock market and each column shows the accuracy with the use of each signal sensitivity. The values can range from 0 to 1, 0 being 0% accuracy and 1 being 100% accuracy. The accuracies are fairly similar. The maximum accuracy of 49.7% can be observed with the NSE stock market while using the 0.1% accuracy, and the minimum value is 41.8 % in the NYSE stock market while using the 1% signal sensitivity. A

significant observation based on the max value of the accuracy is that even in the best-case scenario, the average accuracy of the predictions is below 50%, which indicates that most of the time, the prediction is inaccurate. Also, based on the table, the accuracy is the lowest in the NYSE in general. Only with the signal sensitivity of 0.5%, the accuracy is the same as with NSE and almost as high as with OMXH. On the other hand, the accuracy within NSE is the highest in most of the signals, but with 0.1% signal sensitivity, the accuracy is the highest with OMXH. In conclusion, even though the accuracy seems to be higher with some of the markets, the LSTM model could not be accurate on average with any of the markets. This is not to say that the predictions on individual stocks could not be accurate.

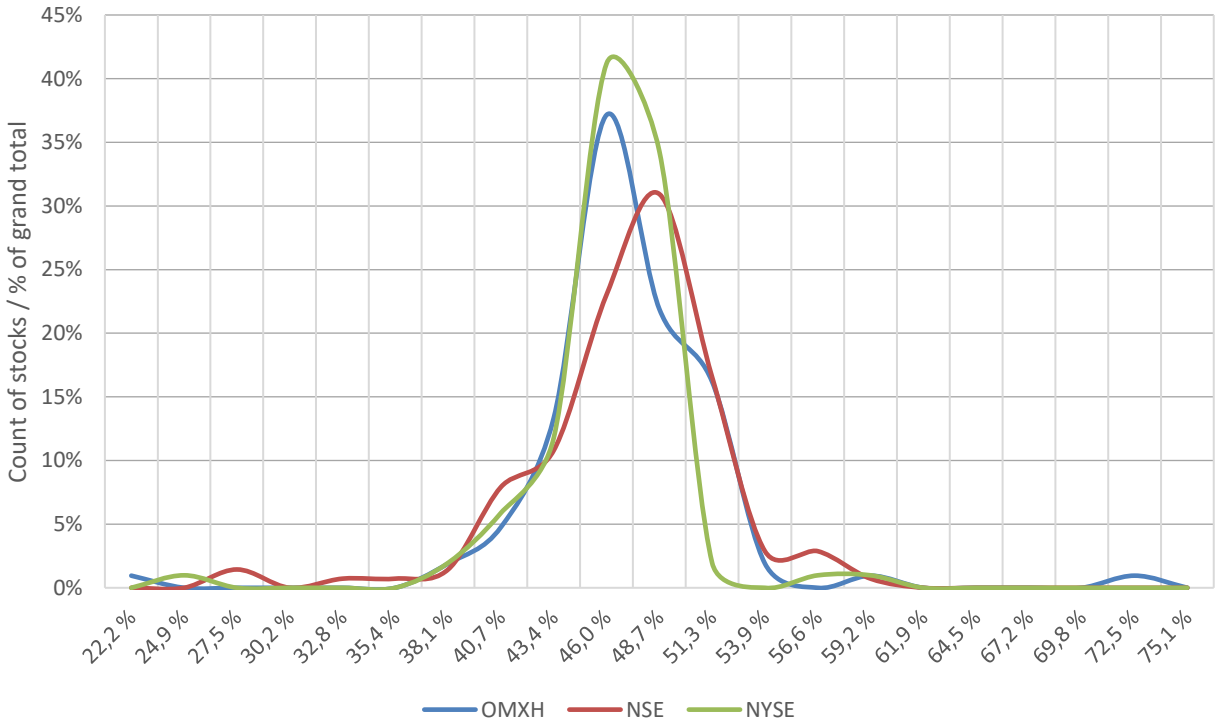

Figure 22 Distribution of LSTM prediction accuracies

The chart above illustrates the distribution of LSTM prediction accuracies for individual stocks across each of the stock markets. Although the LSTM model was not consistently accurate on average, the distribution indicates that, at an individual stock level, LSTM predictions can be considered accurate in some cases. When the distributions are compared across the stock markets, it is clear that there are stocks that are predicted with higher accuracy within NSE and OMXH compared to NYSE, which is in line with the previous observation made based on the table of accuracy metrics. Additionally, the results within NSE and OMXH are more scattered, largely towards the better end of the scale, indicating that more consistent results with NYSE are making NYSE more disadvantageous for the

LSTM model. In the outer ends of the chart, there are some anomalies with especially good or bad accuracies. With OMXH, there is a stock with almost 75 % accuracy, and with NYSE, there is a stock with around 25% accuracy. These are examples of the unpredictability of the model with its accuracy.

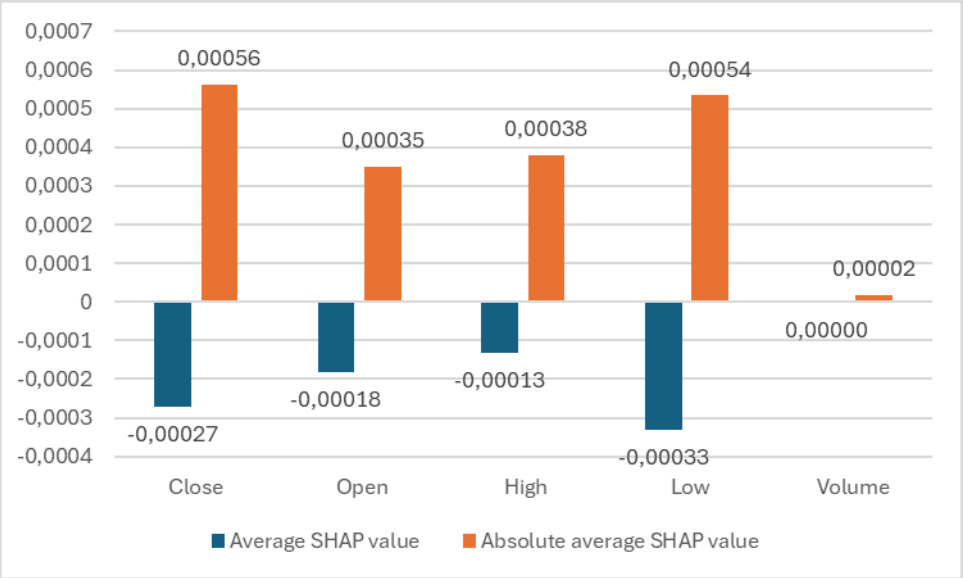

Figure 23 SHAP feature importance

Lastly, we analyse the features or input datapoints of the model. The LSTM model has five data points that it uses as its inputs to model future closing prices. Each input data point has an additive, subtractive or neutral effect on the output. The figure above shows the weight and direction that each feature has on average to the output. The blue bar shows the average direction, whether it's additive, subtractive or neutral on average, and the orange bar shows the weight that it has on average, regardless of the direction of the impact it has. The figure can therefore be used to analyse how the outcome is affected by the changes in the input variables and which features are significant and which aren't. The weight is shown as an absolute value, which is why all of the orange bars are positive. By comparing the absolute average SHAP values, we can observe that the most impactful features are the closing and the intraday low prices. Open and intraday high prices are a bit less impactful, and the volume is the least impactful. These results mean that the LSTM model is assigning more weight to close and low datapoints than the other features in its predictions and almost completely ignoring the volume of the stock. Then by comparing the blue bars, we can observe that, excluding volume, all of the average impacts of the feature for the predictions are negative. Based on this observation, the LSTM model is mostly predicting lower closing prices than they currently are. This indicates that the data has taught the model momentum-based or mean-reversion logic, meaning that the ongoing

trends will change directions eventually, and the trading algorithm, which is based on the model, would benefit from these changes in trend directions.

5 Conclusions

This study aimed to evaluate the effectiveness of the trading algorithms which were based on the LSTM model. There were two hypotheses related to this topic: the LSTM model's predictions can be used to trade individual stocks and outperform the buy-and-hold method, and the LSTM-based allocation methods can be used to outperform the equally weighted portfolio. Based on the initial analysis of the results, even without transaction costs, the first hypothesis can be rejected. The trading algorithm trading individual stocks was not, on average, able to beat the buy-and-hold strategy on any of the stock markets. With some of the individual stocks and on the decreasing OMXH index, the strategy was able to beat the benchmark, but on average, this was not the case. This was most likely due to the low accuracy of LSTM predictions, where most of the predicted price movements were not accurate when compared to the actual price movements afterwards. It also seems that the trading strategy is not able to utilise the ongoing trends in the markets, as with increasing trends the strategy generates less returns than what the buy and hold does, and in contrast, in decreasing markets the returns of the trading strategy are decreasing slower, which is a positive side of the strategy. This being said, the strategy seems to be too inefficient to be beneficial even in decreasing markets, as the outperformance is due to the dampening effect instead of the LSTM predictions. The dampening effect is caused by the difference between the strategy and the buy-and-hold strategy. The buy-and-hold strategy holds a long position throughout the entire observation period, while the LSTM-based trading strategy changes from long to short or no position at all. This causes the dampening effect, as when the trend is increasing and the LSTM predictions are not accurate enough to predict it, the active trading strategy only partly benefits from the trend. The other property of the model, which increases the dampening effect, is logic learned from the learning data. The LSTM model expects the trend to change direction. When the trend stays consistent, the trading strategy is not able to utilise the changing trends that it would benefit from.

The second hypothesis, on the other hand, can initially be accepted according to the results without transaction costs. Both RBA and ERBA were able to outperform the benchmark strategy of the EWP on each of the markets. It seems that although the LSTM model is not able to be accurate in its predictions about individual stocks, it is able to effectively rank the stocks based on their predicted returns. It is therefore more accurate in comparing the stocks instead of predicting them. However, after incorporating transaction costs for the portfolio-level strategies, the hypothesis becomes more nuanced. The analysis showcased the impact of transaction costs on the RBA and ERBA strategies' returns. For RBA to outperform the EWP, the transaction costs needed to be below 0.01% for NSE

and NYSE and below 0.005% for OMXH per transaction. With ERBA, the transaction costs needed to be below 0.01% with NSE and NYSE, but with OMXH, even 0.005% transaction costs made the strategy fall below the EWP. As with the study by Sebastian and Tantia (2024), initial results without transaction costs were promising, but now with the addition of transaction costs, we can observe the limit of costs that the strategy has to be able to operate for it to be more profitable as the passive benchmark of EWP. The hypothesis can therefore only be conditionally accepted, in the case that transaction costs fall below the showcased lines. In future research of this topic, the strategy could consider optimisation around its fees, for example, by less frequent rebalancing.

The second topic of the study was to compare markets with different sizes and economic developmental levels to each other, and whether these properties contributed to the results. The markets were NYSE, OMXH and NSE, and the hypothesis was that NYSE would be the most efficient market due to its size and economic development, thus making it less optimal for trading algorithms and the LSTM model. The trading strategy for individual stocks was not able to turn in excess returns compared to the buy-and-hold strategy on the NYSE or NSE, but on OMXH, it was able to beat the benchmark on average. This difference is most likely caused by the fact that when excess returns are compared to the stock indices themselves, the excess returns increase when the index decreases and when the stock index increases, the excess returns decrease and while the NYSE and NSE are increasing in general, the OMXH is decreasing. This indicates that the trend direction plays a role in the results. Therefore, there isn’t any clear meaningful effect of the market outside of different stock market trends, which would affect the strategy's performance. Even though the trading strategy cannot be reliably analysed for market effects, the accuracy of the LSTM model varies between the markets. The predictions in the NYSE market are the most inaccurate, supporting the hypothesis. The accuracy across the market is also more consistent on NYSE, but the inconsistency on OMXH and the NSE is beneficial as the variance of the accuracy skews towards the better end of the scale, still keeping the average under 50 %. Despite this, there are more stocks within the NSE and OMXH than the NYSE, which have over 50 % accuracy, which would be in support of the hypothesis. The allocation methods, on the other hand, provide contradicting evidence. The largest excess returns, without transaction costs, generated by RBA and ERBA compared to the benchmark EWP are made on the NYSE index. The RBA and ERBA both outperform the EWP on all of the markets, but on the NYSE, the difference is most significant. These observations remain relevant also with transaction costs, and they seem to indicate that the OMXH and NSE are not more optimal for the LSTM model or the trading strategies which the hypothesis suggests. In conclusion, the hypothesis cannot be fully accepted as the results

are not in line with each other, and the most impacts seem to be caused by market conditions instead of market attributes.

The last topic of the thesis was to analyse how the sensitivity affects the performance of the trading strategy that trades individual stocks. The hypothesis was that the more active signal generation would be beneficial for the trading strategy. When comparing the different sensitivity thresholds across the markets, the results are inconsistent. With NYSE, the weaker signal sensitivity or the more passive signal generation is more optimal, although the difference between the sensitivities is the least noticeable compared to the other markets. Within OMXH and NSE, the best-performing signal sensitivities were the more active ones, which would support the hypothesis. The deviation of returns between the sensitivities was still minimal, which does not indicate a great effect of the signal sensitivities on the generated returns. The accuracy analysis on the different signals paints a clearer picture on the matter. The most accurate predictions were made when the signal sensitivity was higher. This was observed on all of the stock markets. Based on this observation, the hypothesis can be at least partly accepted as the more active signal generation benefited the LSTM model in all of the markets and marginally the trading algorithm in OMXH and NSE markets.

The study did not include taxes in the analysis, which limits the practicality of the study. To determine the real-world performance of the trading strategies mentioned in the study, taxes should be considered based on the taxation of the country in which the strategy would be implemented. To improve the results further, different variables of the LSTM model, the rules of the trading algorithms, and transaction cost optimisation could be looked into in more detail. Another limitation comes from the amount of data. The backtesting period, which was only a couple of years, limited the testing to certain market conditions, which can vary heavily from year to year and made it difficult to compare results across markets as their conditions seemed to affect the results more than the markets themselves. Also, due to the positive results from the allocation methods, other allocation methods based on LSTM models could be considered in future studies.

References

Abbracciavento, F., Formentin, S., & Savaresi, S. M. (2022). Data-driven stock trading in financial markets: An adaptive control approach. *International Journal of Control*, *95*(4), 1032–1041. https://doi.org/10.1080/00207179.2020.1837395

Aldridge, I., & Avellaneda, M. (2021). *Big Data Science in Finance*. John Wiley & Sons, Incorporated. http://ebookcentral.proquest.com/lib/kutu/detail.action?docID=6452695

Bachelier, L. (1900). Théorie de la spéculation. *Annales Scientifiques de l'École Normale Supérieure*, *17*, 21–86. https://doi.org/10.24033/asens.476

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLOS ONE*, *12*(7), e0180944. https://doi.org/10.1371/journal.pone.0180944

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4). New York: springer.

Bussmann, N., Giudici, P., Marinelli, D., & Papenbrock, J. (2021). Explainable Machine Learning in Credit Risk Management. *Computational Economics*, *57*(1), 203–216. https://doi.org/10.1007/s10614-020-10042-0

Caldeira, J., & Moura, G. V. (2013). *Selection of a Portfolio of Pairs Based on Cointegration: A Statistical Arbitrage Strategy* (SSRN Scholarly Paper 2196391). Social Science Research Network. https://doi.org/10.2139/ssrn.2196391

Chakravarty, S., & Sarkar, P. (2020). *An Introduction to Algorithmic Finance, Algorithmic Trading and Blockchain*. Emerald Publishing Limited. http://ebookcentral.proquest.com/lib/kutu/detail.action?docID=6287155

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling* (arXiv:1412.3555). arXiv. https://doi.org/10.48550/arXiv.1412.3555

Cummins, M., & Bucca, A. (2011). *Quantitative Spread Trading on Crude Oil and Refined Products Markets* (SSRN Scholarly Paper 1932471). Social Science Research Network. https://doi.org/10.2139/ssrn.1932471

Dubey, R., Babu, A., Jha, R., & Varma, U. (2022). Algorithmic Trading Efficiency and its Impact on Market-Quality. *Asia-Pacific Financial Markets*, *29*. https://doi.org/10.1007/s10690-021-09353-5

Fabbri, M., & Moro, G. (2018). Dow Jones Trading with Deep Learning: The Unreasonable Effectiveness of Recurrent Neural Networks: *Proceedings of the 7th International*

Conference on Data Science, Technology and Applications, 142–153. https://doi.org/10.5220/0006922101420153

Fadlalla, A., & Lin, C.-H. (2001). An Analysis of the Applications of Neural Networks in Finance. *NEURAL NETWORKS IN FINANCE*.

Fama, E. F. (1965). The Behavior of Stock-Market Prices. *The Journal of Business*, *38*(1), 34–105.

Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*, *25*(2), 383–417. https://doi.org/10.2307/2325486

Fama, E. F. (1991). Efficient Capital Markets: II. *The Journal of Finance*, *46*(5), 1575–1617. https://doi.org/10.1111/j.1540-6261.1991.tb04636.x

Fama, E. F., & French, K. R. (1988). Permanent and temporary components of stock prices. *Journal of Political Economy*, *96*(2), 246–273.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*(2), 654–669. https://doi.org/10.1016/j.ejor.2017.11.054

Foltice, B., & Langer, T. (2015). *Profitable Momentum Trading Strategies for Individual Investors* (SSRN Scholarly Paper 2420743). Social Science Research Network. https://papers.ssrn.com/abstract=2420743

Foucault, T., & Moinas, S. (2018). Is Trading Fast Dangerous? In *Global Algorithmic Capital Markets* (pp. 9–27). Oxford University Press. https://doi.org/10.1093/oso/9780198829461.003.0002

Frino, A., Garcia, M., & Zhou, Z. (2020). Impact of algorithmic trading on speed of adjustment to new information: Evidence from interest rate derivatives. *Journal of Futures Markets*, *40*(5), 749–760. https://doi.org/10.1002/fut.22104

Gatev, E., Goetzmann, W. N., & Rouwenhorst, K. G. (2006). Pairs Trading: Performance of a Relative-Value Arbitrage Rule. *Review of Financial Studies*, *19*(3), 797–827. https://doi.org/10.1093/rfs/hhj020

Gomber, P., Arndt, B., Lutat, M., & Uhle, T. (2011). High-Frequency Trading. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.1858626

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2222–2232. https://doi.org/10.1109/TNNLS.2016.2582924

Hindarto, D. (2023). Comparison of RNN Architectures and Non-RNN Architectures in Sentiment Analysis. *Sinkron : Jurnal Dan Penelitian Teknik Informatika*, *7*(4), Article 4. https://doi.org/10.33395/sinkron.v8i4.13048

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Huttunen, J., Jauhiainen, J., Lehti, L., Nyland, A., Martikainen, M., & Lehner, O. (2019). Big data, cloud computing and data science applications in finance and accounting. *ACRN Journal of Finance and Risk Perspectives*, *2019*(8), 16–30.

Kissell, R. (2013). *The Science of Algorithmic Trading and Portfolio Management*. Academic Press.

Koehn, P. (2020). *Neural Machine Translation* (1st ed.). Cambridge University Press. https://doi.org/10.1017/9781108608480

Krauss, C. (2017). STATISTICAL ARBITRAGE PAIRS TRADING STRATEGIES: REVIEW AND OUTLOOK. *Journal of Economic Surveys*, *31*(2), 513–545. https://doi.org/10.1111/joes.12153

Lipton, Z. C. (2015). *A Critical Review of Recurrent Neural Networks for Sequence Learning* (arXiv:1506.00019; Version 2). arXiv. https://doi.org/10.48550/arXiv.1506.00019

Lo, A. W., & MacKinlay, A. C. (1988). Stock Market Prices Do Not Follow Random Walks: Evidence from a Simple Specification Test. *The Review of Financial Studies*, *1*(1), 41–66. https://doi.org/10.1093/rfs/1.1.41

Lo, A. W., & MacKinlay, A. C. (1999). *A Non-Random Walk Down Wall Street*. Princeton University Press. https://www.jstor.org/stable/j.ctt7tccx

López De Prado, M. M. (2020). *Machine Learning for Asset Managers* (1st ed.). Cambridge University Press. https://doi.org/10.1017/9781108883658

Lundberg, S., & Lee, S.-I. (2017). *A Unified Approach to Interpreting Model Predictions* (arXiv:1705.07874). arXiv. https://doi.org/10.48550/arXiv.1705.07874

Lundberg, S. M., Erion, G. G., & Lee, S.-I. (2019). *Consistent Individualized Feature Attribution for Tree Ensembles* (arXiv:1802.03888). arXiv. https://doi.org/10.48550/arXiv.1802.03888

Malkiel, B. G. (1973). *A random walk down Wall Street: Including a life-cycle guide to personal investing*. *1999*.

Malkiel, B. G. (2003). The Efficient Market Hypothesis and Its Critics. *Journal of Economic Perspectives*, 59–82.

Martin, R. J. (2023). *Design and analysis of momentum trading strategies* (arXiv:2101.01006). arXiv. https://doi.org/10.48550/arXiv.2101.01006

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://doi.org/10.1007/BF02478259

Misheva, B. H., Osterrieder, J., Hirsa, A., Kulkarni, O., & Lin, S. F. (2021). *Explainable AI in Credit Risk Management* (arXiv:2103.00949). arXiv. https://doi.org/10.48550/arXiv.2103.00949

Nelson, D. M. Q., Pereira, A. C. M., & De Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. *2017 International Joint Conference on Neural Networks (IJCNN)*, 1419–1426. https://doi.org/10.1109/IJCNN.2017.7966019

Poterba, J. M., & Summers, L. H. (1988). Mean reversion in stock prices: Evidence and Implications. *Journal of Financial Economics*, *22*(1), 27–59. https://doi.org/10.1016/0304-405X(88)90021-9

Pricope, T.-V. (2021). *Deep Reinforcement Learning in Quantitative Algorithmic Trading: A Review* (arXiv:2106.00123). arXiv. https://doi.org/10.48550/arXiv.2106.00123

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519

Sebastian, A., & Tantia, V. (2024). Deep Learning for Stock Price Prediction and Portfolio Optimization. *International Journal of Advanced Computer Science and Applications (IJACSA)*, *15*(9), Article 9. https://doi.org/10.14569/IJACSA.2024.0150995

Seyfert, R. (2018). Automation and affect: A study of algorithmic trading. In *Affect in Relation*. Routledge.

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied Soft Computing*, *90*, 106181. https://doi.org/10.1016/j.asoc.2020.106181

Teall, J. L. (2018). Chapter 1—Introduction to Securities Trading and Markets. In J. L. Teall (Ed.), *Financial Trading and Investing (Second Edition)* (pp. 1–31). Academic Press. https://doi.org/10.1016/B978-0-12-811116-1.00001-5

Treleaven, P., Galas, M., & Lalchand, V. (2013). Algorithmic trading review. *Communications of the ACM*, *56*(11), 76–85. https://doi.org/10.1145/2500117

Tsinaslanidis, P., & Guijarro, F. (2021). What makes trading strategies based on chart pattern recognition profitable? *Expert Systems*, *38*(5), e12596. https://doi.org/10.1111/exsy.12596

Venkatarathnam, N., Goranta, L. R., Kiran, P. C., Raju, B. P. G., Dilli, S., Basha, S. M., & Kethan, M. (2024). An Empirical Study on Implementation of AI & ML in Stock Market Prediction.

Indian Journal of Information Sources and Services, *14*(4), 165–174. https://doi.org/10.51983/ijiss-2024.14.4.26

Wang, J., Hong, S., Dong, Y., Li, Z., & Hu, J. (2024). Predicting Stock Market Trends Using LSTM Networks: Overcoming RNN Limitations for Improved Financial Forecasting. *Journal of Computer Science and Software Applications*, *4*(3), Article 3. https://doi.org/10.5281/zenodo.12200708

Yadav, A., Jha, C. K., & Sharan, A. (2020). Optimizing LSTM for time series prediction in Indian stock market. *Procedia Computer Science*, *167*, 2091–2100. https://doi.org/10.1016/j.procs.2020.03.257

Zhang, X., & Tan, Y. (2018). Deep Stock Ranker: A LSTM Neural Network Model for Stock Selection. In Y. Tan, Y. Shi, & Q. Tang (Eds.), *Data Mining and Big Data* (pp. 614–623). Springer International Publishing. https://doi.org/10.1007/978-3-319-93803-5_58

The code for the empirical part of the study is documented in the following GitHub repository: https://github.com/veehark/lstm-trading-algorithm

Appendices

Appendix 1 Stock basket: Nasdaq Helsinki

Nasdaq Helsinki

Ticker	Name
NOKIA.HE	Nokia Oyj
KNEBV.HE	KONE Oyj
STERV.HE	Stora Enso Oyj
FORTUM.HE	Fortum Oyj
TIETO.HE	TietoEVRY Oyj
TYRES.HE	Nokian Renkaat Oyj
METSB.HE	Metsä Board Oyj
KESKOB.HE	Kesko Oyj
HUH1V.HE	Huhtamäki Oyj
ELISA.HE	Elisa Oyj
TELIA1.HE	Telia Company AB (publ)
ORNBV.HE	Orion Oyj
NESTE.HE	Neste Oyj
UPM.HE	UPM-Kymmene Oyj
NDA-FI.HE	Nordea Bank Abp
WRT1V.HE	Wärtsilä Oyj Abp
METSO.HE	Metso Oyj
SAMPO.HE	Sampo Oyj
KCR.HE	Konecranes Plc
OUT1V.HE	Outokumpu Oyj
CGCBV.HE	Cargotec Corporation

Appendix 2 Stock basket: New York Stock Exchange

New York Stock Exchange

Ticker	Name
SCX	Starrett Company
ACCO	ACCO Brands Corporation
WMB	The Williams Companies, Inc.
BCO	The Brink's Company
BCE	BCE Inc.
WLK	Westlake Corporation
WLY	John Wiley & Sons, Inc.
BCC	Boise Cascade Company
BBY	Best Buy Co., Inc.
SCS	Steelcase Inc.
SEE	Sealed Air Corporation
BBW	Build-A-Bear Workshop, Inc.
BCH	Banco de Chile
BSAC	Banco Santander-Chile
CMRE	Costamare Inc.

Appendix 3 Stock basket: National Stock Exchange of India

National Stock Exchange of India

Ticker	Name
TATASTEEL.NS	Tata Steel Limited
TATACONSUM.NS	Tata Consumer Products Limited
RELIANCE.NS	Reliance Industries Limited
LT.NS	Larsen & Toubro Limited
TCS.NS	Tata Consultancy Services Limited
KOTAKBANK.NS	Kotak Mahindra Bank Limited
HINDALCO.NS	Hindalco Industries Limited
LTIM.NS	LTIMindtree Limited
MARUTI.NS	Maruti Suzuki India Limited
ULTRACEMCO.NS	UltraTech Cement Limited
BAJFINANCE.NS	Bajaj Finance Limited
WIPRO.NS	Wipro Limited
BRITANNIA.NS	Britannia Industries Limited
BAJAJ-AUTO.NS	Bajaj Auto Limited
SHRIRAMFIN.NS	Shriram Finance Limited
BAJAJFINSV.NS	Bajaj Finserv Ltd.
ITC.NS	ITC Limited
HEROMOTOCO.NS	Hero MotoCorp Limited
CIPLA.NS	Cipla Limited
NTPC.NS	NTPC Limited
ADANIENT.NS	Adani Enterprises Limited
ONGC.NS	Oil and Natural Gas Corporation Limited
NESTLEIND.NS	Nestlé India Limited
TITAN.NS	Titan Company Limited
APOLLOHOSP.NS	Apollo Hospitals Enterprise Limited
BHARTIARTL.NS	Bharti Airtel Limited